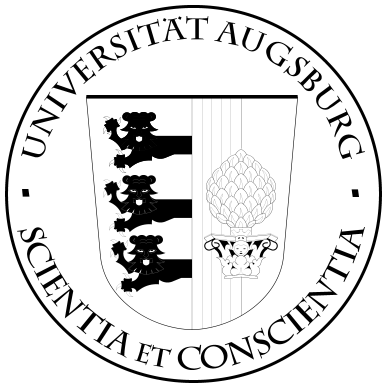
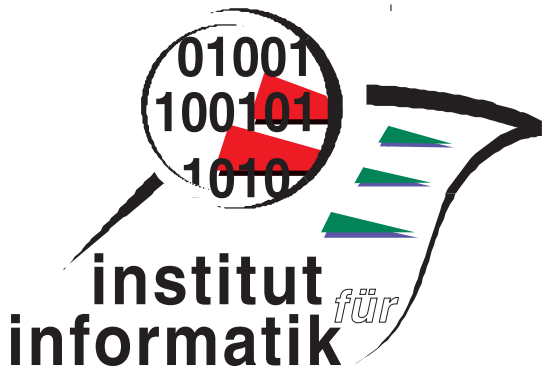
Universität Augsburg



Einsatz von Sprungvorhersagetechniken zur Kontextvorhersage in ubiquitären Systemen

Jan Petzold

Report 2003-4 März 2003



Institut für Informatik
D-86135 Augsburg

Einsatz von Sprungvorhersagetechniken zur Kontextvorhersage in ubiquitären Systemen

Jan Petzold

Zusammenfassung

Ubiquitäre Computersysteme werden nach den Generationen der Großrechner und der Personal Computer als die dritte Rechnergeneration gesehen. Die Benutzer sind von vielen kleineren Rechnern umgeben, die - für den Menschen unsichtbar - eigenständig Unterstützungen im täglichen Leben bieten. Um sich in hohem Maße auf den Menschen einzustellen, werden Informationen der Umgebung benötigt, welche als Kontext bezeichnet werden. Eine weiterreichende Unterstützung liegt darin, dem Menschen Entscheidungen abzunehmen, die aus Gewohnheit resultieren. Dazu wird es zwingend Kontexte vorherzusagen, so dass das System in einem vom Benutzer gewünschtem Maße proaktiv handeln kann.
In der vorliegenden Arbeit wird ein Lösungsansatz für die Kontextvorhersage in ubiquitären Systemen präsentiert. Hierbei werden Sprungvorhersagetechniken aus der Prozessorarchitektur dahingehend untersucht, wie diese sich auf die Kontextvorhersage abbilden lassen. Es werden die dynamischen einstufigen Prädiktoren wie der Ein-Bit-Prädiktor und der Zwei-Bit-Prädiktor, sowie die zweistufig adaptiven Prädiktoren übertragen. In der Evaluierung wird ausgewertet, ob die sehr guten Ergebnisse der Sprungvorhersage auch in der realen Welt Bestand haben.

Inhaltsverzeichnis

1 Einleitung

In dieser Arbeit soll der Einsatz von Mechanismen für Kontextvorhersagen in Ubiquitären Systemen untersucht werden. Die Kombination verschiedenster bekannter Informationen soll dazu benutzt werden, aus einem augenblicklichen Kontext heraus zukünftige Kontexte vorherzusagen. Ziel ist es, ein lernfähiges System zu entwickeln, welches sich Fehlschätzungen merkt und seine Trefferwahrscheinlichkeit erhöhen kann.

Die Fähigkeit Kontexte vorherzusagen, eröffnet im Bereich der Ubiquitären Systeme Möglichkeiten, dem Menschen Entscheidungen abzunehmen, die aus Gewohnheiten resultieren, welche sich in Verhaltensmustern ausdrücken. Kontextvorhersage kann damit künftig eine unerlässliche Unterstützung im täglichen Leben bieten. Weiterhin bietet die Kontextvorhersage einem ubiquitären System die Möglichkeit proaktiv zu handeln. Dieses proaktive Handeln kann beispielsweise für die Initialisierung eines Geräte oder das Umschalten zwischen Geräten von Bedeutung sein.

Als mögliche Vorhersagetechniken fällt das Augenmerk auf Neuronale Netze, Bayes'sche Netze sowie Markov-Ketten als Verfahren des Maschinellen Lernens. Diese Verfahren sollen auch zu einem späteren Zeitpunkt untersucht werden. In dieser Arbeit werden Methoden zur Sprungvorhersage aus dem Bereich der Prozessorarchitektur auf ihre Einsetzbarkeit hin betrachtet.

Für die Kontextvorhersage muss zunächst aber der Begriff Kontext im Bereich Ubiquitärer Systeme hergeleitet und abgegrenzt werden. Auch soll der Begriff Kontextvorhersage genauer spezifiziert werden.

2 Kontext

2.1 Definition

Ein Kontext in ubiquitären Systemen beschreibt den Zustand, der durch Auswertung von Informationen der Umgebung entsteht, in der sich der Benutzer oder ein System gerade befinden. Diese Zustandsdaten können über verschiedene Sensoren erfasst werden.

Der Begriff Kontext spielt in allen Bereichen der Informatik eine große Rolle. In ubiquitären Systemen gibt es dementsprechend schon viele Modelle für Kontext. Bereits ein einzelner Sensorwert (z.B. eine bestimmte Lumenzahl) kann einen einfachen Kontext herstellen (z.B. Licht an bzw. aus) [2].

Oft wird der Begriff Kontext in erster Linie nur für die Beschreibung der physischen Umgebung eines Benutzers oder Systems verwendet. In diesem Fall wird dann vom physischen Kontext (*physical context*) gesprochen [5]. In anderen Modellen wird zusätzlich ein situationsbezogener Kontext (*situational context*) mit eingebunden [6, 12]. Hiermit wird eine Situation beschrieben, in der sich der Benutzer oder das System augenblicklich befinden. Diese Situationen können meist nicht durch die reine Erfassung physischer Daten ermittelt werden, sondern es müssen verschiedene Sensordaten analysiert und kombiniert werden.

Im wohl meist zitiertem Kontextmodell der ubiquitären Systeme, dem *Context Toolkit* [11] wird Kontext folgendermaßen definiert: Kontext ist die Umgebungsinformation, die ein Teil der Betriebsumgebung einer Anwendung ist und die durch die Anwendung abgefragt werden kann.

2.2 Modellierung

Das in dieser Arbeit verwendete Kontextmodell [1] wird wie folgt aufgebaut: Im einfachsten Fall ist ein Kontext eine einzelne Merkmalsinformation. Betrachtet man zum Beispiel den Kontext des Lichtes, so kann dieser aus der einzelnen Information „Licht an“ oder „Licht aus“ bestehen. Durch

die sinnvolle Zusammenfassung mehrerer solcher einfachen Kontexte entstehen neue zusammengesetzte Kontexte. Zum Beispiel kann man zu der Information „Licht an“ bzw. „Licht aus“ noch die Leistung des Lichtes hinzunehmen, um den Kontext des Lichtes zu beschreiben. Durch geeignete Kombination zusammengesetzter Kontexte können dann komplexe Kontexte gebildet werden. Im betrachteten Beispiel können der Kontext des Lichtes und die Kontexte weiterer Geräte im Kontext des Systems vereint werden.

Gegeben sei eine Menge von Bezeichner L sowie eine Menge von Zuständen S. Gesucht ist die Menge der Kontexte C. Ein Kontext $c \in C$ kann dann eine der folgenden Form haben:

- Einfacher Kontext

  $$c := (L, S)$$

  Beispiel:

  ```
  Stromkontext = (Strom, an)
  ```

- Zusammengesetzter Kontext

  $$c := (L, (L, S)^+)$$

  Beispiel:

  ```
  Lichtkontext = (Licht, (Strom, an),
                         (Leistung, 60))
  ```

- Komplexer Kontext

  $$c := (L, C^+)$$

  Beispiel:

  ```
  Raumkontext = (Raum, (Licht, (Strom, an),
                               (Leistung, 60)),
                       (Radio, (Strom, an),
                               (Lautstärke, 12),
                               (Sender, Bayern 3)))
  ```

2.3 Eigenschaften

Kontexte in Ubiquitären Systemen zeichnen sich durch folgende Eigenschaften aus:

- *dynamisch*

  Ein situationsbezogener Kontext kann sich durch Ortswechsel oder neue Umgebungsbedingungen ändern. Neue Informationen sollen in einen Kontext aufgenommen bzw. vorhandene Informationen aus einem Kontext gelöscht werden können, d.h. Kontexte sollen nicht statisch festdefiniert werden, sondern leicht änderbar sein.

- *zeitabhängig*

  Eine grundlegende Eigenschaft ist die Zeitabhängigkeit. Hierbei ist zu unterscheiden, dass sich auf der einen Seite ein Kontext mit der Zeit ändern kann ohne wieder in den ursprünglichen Zustand zurückzukehren. Auf der anderen Seite kann ein Kontext seinen Zustand periodisch immer wieder ändern. Die periodische Anpassung wird für die Kontextvorhersage von großer Bedeutung sein.

- *ortsabhängig*

  Auch die Ortsabhängigkeit ist sehr wichtig. Ein mobiles System wird an verschieden Orten entsprechend verschiedene Kontexte haben.

- *untereinander korreliert*

  Kontexte hängen voneinander in dem Sinn ab, dass eine Änderung eines Kontextes einen anderen beeinflusst, so dass dieser sich auch ändern kann.

- *zusammensetzbar*

  Kontexte sollen zu komplexeren Kontexten wie vorab schon beschrieben zusammensetzbar sein.

- *unscharf*

  Kontexte können klar definiert oder aber auch unscharf definiert sein. Insbesondere kann ein situationsbezogener Kontext unscharf sein in dem Sinn, dass die Sensorwerte, welche die Situation umschreiben, ein großes Spektrum haben können.

3 Kontextvorhersage

3.1 Definition

In Kombination mit verschiedensten bekannten Informationen sollen aus dem augenblicklichen Kontext und vergangenen Kontexten heraus zukünftige Kontexte vorhergesagt werden. Formell ausgedrückt, soll der Kontext c_t aus den vergangenen n Kontexten $c_{t-1}, \ldots, c_{t-n}$ vorhergesagt werden. Seien $c_{t_1}, \ldots, c_{t_k}$ die möglichen nächsten Kontexte, die eintreten können, dann soll der Kontext mit der größten Wahrscheinlichkeit

$$P(c_{t_i}|c_{t-1}, \ldots, c_{t-n}) \quad i = 1, \ldots, k$$

bestimmt werden. Aus einem augenblicklichen Kontext den nächsten Kontext vorherzusagen kann auch bedeuten, vorherzusagen wie sich die Zustände der Informationen ändern, die dieser Kontext umfasst, d.h. es tritt kein neuer Kontext auf, sondern es ändern sich nur die Werte des Kontextes.

Der Begriff *context prediction* wird bei Gellersen et al [6] so verwendet, dass aus verschiedenen Informationen, wie Sensorendaten, ein augenblicklicher situationsbezogener Kontext, welcher nicht direkt aus Sensoren gewonnen werden kann, vorhergesagt wird. Aus diesem Grund sollte die hier beschriebene Kontextvorhersage im Englischen auch präziser mit *next context prediction* bezeichnet werden.

Im Folgenden werden Mechanismen untersucht, mit denen Kontexte vorhergesagt werden können. Ziel ist es, ein lernfähiges System zu entwickeln, welches sich Fehlschätzungen merkt und seine Trefferwahrscheinlichkeit erhöhen kann. An so ein System müssen verschiedene Anforderungen gestellt werden, die im nächsten Abschnitt besprochen werden.

3.2 Anforderungen

An einen Kontextvorhersage-Algorithmus werden folgende Anforderungen gestellt:

- Er soll schnell lernen und auch schnell umlernen (dynamisch).
- Ein Kernpunkt ist die Zeitabhängigkeit. Hierbei muss insbesondere unterschieden werden, wann wird eine Vorhersage gemacht und wann tritt der neue Kontext ein.
- Neben der Zeitabhängigkeit ist auch die Ortsabhängigkeit zu berücksichtigen.
- Er soll eine Relevanz der Informationen beachten, d.h. er soll die verschiedenen Informationen gewichtet einbeziehen.

Ein System, in welchem der Algorithmus eingebettet ist, sollte zusätzlich noch folgende Anforderungen erfüllen:

- Neue Kontexte sollen mit eingebaut werden, der Algorithmus soll eine Art offenes System darstellen.
- Wenn ein Kontext falsch vorhergesagt wurde, gibt es Situationen, in denen ein Rückrollen der aufgrund der Vorhersage durchgeführten Aktionen notwendig ist. Zum Beispiel wird aufgrund einer falschen Vorhersage das Licht eingeschaltet, wer schaltet es nun wieder aus?
- Auch muss er die Korreliertheit berücksichtigen. Zum einen ist die Korreliertheit zwischen den Kontexten zu beachten. Zum anderen entsteht Korreliertheit unter den Vorhersagen, d.h. die Vorhersagen können voneinander abhängig sein.
- Da die Domänen der Informationen alle möglichen Wertigkeiten annehmen können, z.B binär, diskret, kontinuierlich usw., muss der Algorithmus eine mehrwertige Logik unterstützen.

3.3 Ansätze

Unser Ansatz liegt im Einsatz von Sprungvorhersagetechniken (*branch prediction*) aus dem Bereich der Prozessorarchitektur [3]. Hierbei soll untersucht werden, wie sich diese Verfahren auf die Kontextvorhersagen abbilden lassen, und ob dabei die aus der Prozessorarchitektur bekannte große Trefferwahrscheinlichkeit erhalten bleibt. Andere Vorhersagetechniken basieren auf Neuronalen Netzen [9], Bayes'schen Netzen [8] und Markov-Ketten [7].

Weiterhin zeichnen sich die Algorithmen der Sprungvorhersage durch ihren geringen Speicheraufwand aus. Sie sind sehr einfach und schnell, so dass kein großer Aufwand an Rechenleistung nötig ist. Diese Eigenschaft ist für die energiesparenden und langsamen ubiquitären Geräte von großer Bedeutung.

Im Folgendem werden zwei Beispiele vorgestellt, die zeigen wie die Methoden der Sprungvorhersage auf die Kontextvorhersage projiziert werden können.

Erlernen einer einzelnen Information

Die einfachste Vorhersage besteht darin, einen durch den Benutzer in der Vergangenheit geänderten Kontext vorherzusagen.

Beispiel: Eine Person betritt einen Raum täglich zur gleichen Zeit und schaltet das Licht ein. Das System soll feststellen, dass die Person die Aktion mehrmals ausgeführt, und schaltet somit beim nächsten Betreten des Raumes durch diese Person zu dem bestimmten Zeitpunkt das Licht vor dem Eintreten ein. Umgekehrt schaltet die Person nun beim Betreten mehrmals das Licht hintereinander wieder aus, so soll auch das System das Licht beim nächsten Betreten nicht mehr anschalten.

Lösung: 2-Bit-Prädiktor mit Sättigung [3].

Erkennen von Bewegungsabläufen

Es sollen Muster erkannt werden, die durch die Bewegung von Personen in Gebäuden entstehen.

Beispiel: Ein Angestellter betritt morgens immer zuerst sein Büro anschließend geht er immer in die Küche, um Kaffee zu kochen bzw. zu holen, dann geht er wieder in sein Büro.

Lösung: Zweistufig adaptiver Prädiktor [3].

Um die Techniken der Sprungvorhersage auf Kontexte anzuwenden, sollen im nächsten Abschnitt diese Techniken kurz beschrieben werden.

4 Sprungvorhersage

4.1 Grundlagen

Für heutige und zukünftige Mikroprozessoren ist eine exzellente Behandlung von Sprungbefehlen sehr wichtig. Deshalb sind die Sprungvorhersagetechniken in den letzten 20 Jahren sehr intensiv erforscht worden.

Die Gesamtleistung einer Sprungvorhersage hängt zum einen von der Genauigkeit der Vorhersage und zum anderen von den Kosten für das Rückrollen bei einer Fehlspekulation ab. Die Genauigkeit kann durch eine qualitativ bessere Sprungvorhersagetechnik erhöht werden. Der Einsatz größerer Informationstabellen über die bisherigen Sprungverläufe führt bei diesen Techniken auch zu weniger Fehlspekulationen [3].

4.2 Dynamische Sprungvorhersagetechniken

Bei der dynamischen Sprungvorhersagetechnik wird die Entscheidung über die Spekulationsrichtung eines Sprungs in Abhängigkeit vom bisherigen Programmablauf getroffen. Die Sprungverläufe werden beim Programmablauf in Sprungverlaufstabellen gesammelt und auf der Grundlage der Tabelleneinträge werden aktuelle Sprungvorhersagen getroffen. Bei Fehlspekulationen werden die Tabellen per Hardware abgeändert.

Als dynamische Sprungvorhersagetechniken kommen heute neben dem einfachen Zwei-Bit-Prädiktor zunehmend zweistufig adaptive Prädiktoren und Hybrid-Prädiktoren zum Einsatz. Diese Techniken werden in den nachfolgenden Abschnitten beschrieben (siehe auch [3]).

4.3 Ein- und Zwei-Bit-Prädiktoren

Die einfachste dynamische Sprungvorhersagetechnik ist der **Ein-Bit-Prädiktor**, der für jeden Sprungbefehl die zwei Zustände „genommen“ oder „nicht genommen“ in einem Bit speichert. Diese Zustände beziehen sich dabei immer auf die zeitlich letzte Ausführung des Sprungbefehls. Die Zustände eines Ein-Bit-Prädiktors sind in Abb. 1 dargestellt. Dabei steht T für „genommen“ (*taken*) und NT für „nicht genommen“ (*not taken*).

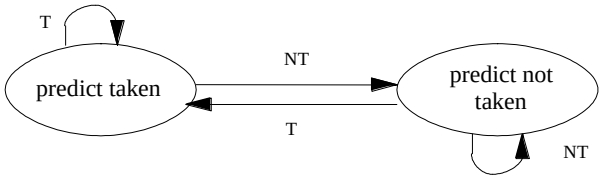


Abbildung 1: Die Zustände des Ein-Bit-Prädiktors

Der Ein-Bit-Prädiktor benötigt eine Sprungverlaufstabelle, die mit einem Teil der Sprungbefehlsadresse adressiert wird und pro Eintrag ein Bit enthält, das die Sprungvorhersage bestimmt. Ist dieses Bit gesetzt, wird der Sprung als „genommen“ vorhergesagt, und wenn es gelöscht ist, als „nicht genommen“. Im Falle einer Fehlspekulation wird das Bit invertiert und damit die Richtung der Vorhersage umgekehrt.

Der Ein-Bit-Prädiktor sagt jeden Sprung am Ende einer Schleifeniteration richtig voraus, solange die Schleife iteriert wird. Der Prädiktor des Sprungbefehls steht auf „genommen“ (T). Wird die Schleife verlassen, so ergibt sich eine falsche Vorhersage und damit eine Invertierung des Vorhersagebits des Sprungbefehls, auf „nicht genommen“ (NT). Damit kommt es in geschachtelten

Schleifen jedoch in der inneren Schleife zu einer weiteren falschen Vorhersage. Beim Wiedereintritt in die innere Schleife steht am Ende der ersten Iteration der inneren Schleife die Vorhersage noch auf NT. Die zweite Iteration wird damit falsch vorhergesagt, denn erst ab dieser zweiten Iteration steht der Prädiktor des Sprungbefehls wieder auf „genommen“ (T). Mit einem Zwei-Bit-Prädiktor wird bei geschachtelten Schleifen eine dieser zwei Fehlvorhersagen vermieden.

Beim **Zwei-Bit-Prädiktor** werden für die Zustandscodierungen der bedingten Sprungbefehle zwei Bits pro Eintrag in der Sprungverlaufstabelle verwendet. Damit ergeben sich die vier Zustände „sicher genommen“ (*strongly taken*), „vielleicht genommen“ (*weakly taken*), „vielleicht nicht genommen“ (*weakly not taken*) und „sicher nicht genommen“ (*strongly not taken*). Befindet sich ein Sprungbefehl in einem „sicheren“ Vorhersagezustand, so sind zwei aufeinander folgende Fehlspekulationen nötig, um die Vorhersagerichtung umzudrehen. Damit kommt es bei inneren Schleifen einer Schleifenschachtelung nur beim Austritt aus der Schleife zu einer falschen Vorhersage.

Es gibt zwei Ausprägungen des Zwei-Bit-Prädiktors, die sich in der Definition der Zustandsübergänge unterscheiden. Das Schema mit einem **Sättigungszähler** ist in Abb. 2 dargestellt. Die zweite, **Hysteresezähler** genannte Variante verdeutlicht Abb. 3.

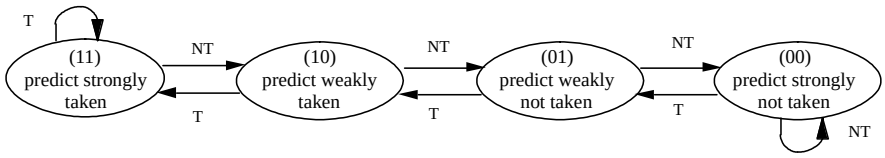


Abbildung 2: Die Zustände des Zwei-Bit-Prädiktors mit Sättigungszähler

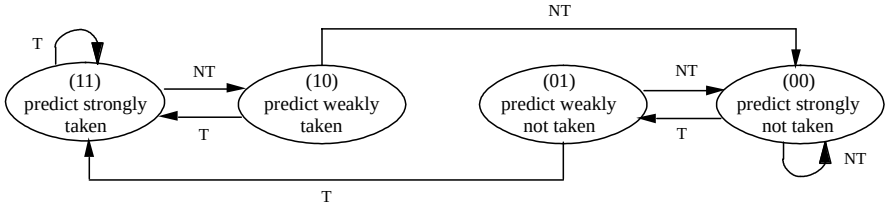


Abbildung 3: Die Zustände des Zwei-Bit-Prädiktors mit Hysteresezähler

Der Zwei-Bit-Prädiktor mit Sättigungszähler erhöht jedesmal, wenn der Sprung genommen wurde, den Zähler und erniedrigt ihn, falls er nicht genommenen wurde. Durch die Sättigungsarithmetik fällt der Zähler nie unter null (00) oder wird größer als drei (11). Das höchstwertige Bit gibt die Richtung der Vorhersage an.

Die zweite Variante, die Hysteresemethode, unterscheidet sich von der des Sättigungszählers dadurch, dass direkt von einem „unsicheren“ (*weakly*) Zustand in den „sicheren“ (*strongly*) Zustand der entgegengesetzten Richtung gewechselt wird. Damit kommt man von einem sicheren Zustand in den anderen sicheren Zustand durch zwei Fehlspekulationen.

Die Technik der Zwei-Bit-Prädiktoren lässt sich leicht auf n Bits erweitern. Es zeigte sich jedoch, dass dabei für die Sprungvorhersage so gut wie keine Verbesserungen mehr erzielbar sind.

4.4 Korrelationsprädiktoren

Die Zwei-Bit-Prädiktoren ziehen für eine Vorhersage immer nur den Verlauf des Sprungs selbst in Betracht. Die Beziehungen zwischen verschiedenen Sprüngen werden nicht berücksichtigt. Untersuchungen haben jedoch gezeigt, dass bei Auswertung dieser Beziehungen eine bessere Sprungvorhersage durchgeführt werden kann.

Die von Pan et al. [10] entwickelten **Korrelationsprädiktoren** berücksichtigen neben der eigenen Vergangenheit eines Sprungbefehls auch die Historie benachbarter, im Programmlauf vorhergegangener Sprünge. Korrelationsprädiktoren erzielen gewöhnlich bei ganzzahlintensiven Programmen eine höhere Trefferrate als die Zwei-Bit-Prädiktoren. Sie benötigen dabei nur wenig mehr Hardware.

4.5 Zweistufig adaptive Prädiktoren

Die zweistufig adaptiven Prädiktoren wurden von Yeh und Patt [13] etwa zur gleichen Zeit wie die eng verwandten Korrelationsprädiktoren entwickelt. Wie der Korrelationsprädiktor ist ein zweistufig adaptiver Prädiktor aus zwei Tabellenebenen aufgebaut, wobei der Eintrag in der ersten Tabelle dazu dient, die Vorhersagebits auf der zweiten Tabellenebene zu selektieren.

Die grundlegenden, zweistufig adaptiven Prädiktorschemata benutzen ein einziges globales Sprungverlaufsregister von k Bit Länge als Index, um einen Eintrag in einer Sprungverlaufstabelle mit Zwei-Bit-Zählern zu adressieren. Das globale BHR wird nach jeder Ausführung eines (bedingten) Sprungbefehls geändert, d.h., der Inhalt des als Schieberegister organisierten BHR wird bitweise von rechts nach links geschoben und der Bitwert '1' für einen genommenen und '0' für einen nicht genommenen Sprung nach jeder Sprungentscheidung an der rechten Bitposition eingefügt. Damit wird, wie bei den Korrelationsprädiktoren, nicht nur der Verlauf eines einzelnen Sprungbefehls, sondern die Aufeinanderfolge von Sprungbefehlen für die Sprungvorhersage einbezogen. Alle zweistufig adaptiven Prädiktoren, die ein globales BHR benutzen, gehören zu den **globalen Verlaufsschemata** und entsprechen den Korrelationsprädiktoren.

Im einfachsten Fall gibt es ein einziges globales BHR (als G bezeichnet) und eine einzige globale PHT (als g bezeichnet). Dieser einfache Prädiktor wird **GAg-Prädiktor** genannt. Alle PHT-Implementierungen der zweistufig adaptiven Prädiktoren von Yeh und Patt benutzen 2-Bit-Prädiktoren. Eine Implementierung eines GAg-Prädiktors mit einem 4 Bit langen BHR (deshalb als GAg(4) bezeichnet) wird in Abb. 4 gezeigt.

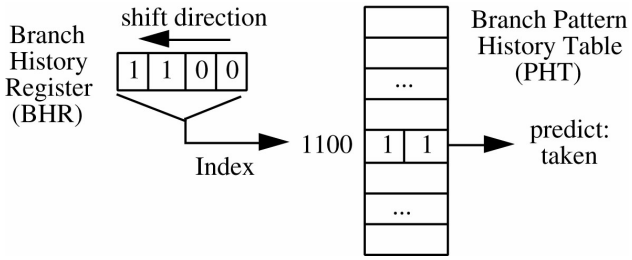


Abbildung 4: Implementierung eines GAg(4)-Prädiktors

Yeh und Patt [13] definieren acht weitere zweistufig adaptive Prädiktoren, auf die hier nicht weiter eingegangen wird (siehe [3, 13] für eine Erläuterung).

4.6 Sprungvorhersage mit Markov-Ketten

Um einen zweistufig adaptiven Prädiktor auf andere Anwendungen abzubilden, kann ein mathematisches Modell von Vorteil sein. Dazu soll die Arbeit von Chen et al [4] betrachtet werden. Hier werden die zweistufig adaptiven Prädiktoren mit Verfahren aus der Datenkomprimierung verglichen. Dabei wurde festgestellt, dass die zweistufig adaptiven Prädiktoren eine Approximationen der *Prediction by Partial Matching* (PPM) sind. Der PPM-Algorithmus beruht auf dem Einsatz von Markov-Ketten (siehe Anhang). Der Unterschied zu den zweistufig adaptiven Prädiktoren der Sprungvorhersage ist der, dass in der zweiten Stufe kein 2-Bit-Prädiktor verwendet wird. Hier wird

das Auftreten, ob der Sprung genommen wurde oder nicht, jeweils hochgezählt. Vorhergesagt wird dann der Zustand mit der größten Häufigkeit.

5 Abbildung der Sprungvorhersagetechniken auf die Kontextvorhersage

Hier sollen Techniken der Sprungvorhersage auf die Kontextvorhersage übertragen werden. Diese Abbildung soll beispielhaft an der Vorhersage von Bewegungsabläufen betrachtet werden: Eine Person bewegt sich durch ein Haus mit mehreren Räumen. Im ersten Schritt soll unabhängig von der Zeit vorhergesagt werden, in welchen Raum eine Person als nächstes geht. Vereinfacht soll zunächst nur eine einzelne Person betrachtet werden. Solange die Bewegungsabläufe mehrerer Personen nicht von einander abhängen, bedeutet dies auch keine Einschränkung.

Folgendermaßen sollen die Prädiktoren der Sprungvorhersage auf die Kontextvorhersage übertragen werden: Der Übergang von einem Raum in seine Nachbarräume wird als Sprungbefehl interpretiert. Hat der Raum nur zwei Nachbarräume, stellt ein Nachbarraum den Zustand „genommen“ und der andere den Zustand „nicht genommen“ dar. Bei mehr als zwei Nachbarräumen müssen weitere Zustände hinzugenommen werden.

5.1 Lokale einstufige Kontextprädiktoren

Zunächst werden die einstufigen Kontextprädiktoren beschrieben. Es ist zu beachten, dass diese Prädiktoren nur lokal untersucht werden. Es wäre nicht sinnvoll, global den nächsten Raum vorherzusagen, da in einem Gebäude nicht jeder Raum von jedem anderen erreichbar ist. Deshalb soll es für jeden Raum jeweils einen dieser Prädiktoren geben. Die einstufigen Kontextprädiktoren werden auch nicht initialisiert, d.h. es gibt keine Startzustände, da es besser ist keine als eine falsche Vorhersage zu treffen.

5.1.1 1-State-Kontextprädiktor

Beim Verlassen eines Raumes R wird der Zielraum Z gespeichert. Kommt die Person wieder in den Raum R wird nun als nächster Raum der Raum Z vorhergesagt. Der Prädiktor hat für jeden Raum einen Zustand, in dem dieser vorhergesagt wird. Aus diesem Grund wurde auch die Bezeichnung 1-State-Kontextprädiktor gewählt. Im weiteren wird der Prädiktor auch verkürzt 1-State-Prädiktor genannt.

- Bei *zwei* Nachbarräumen ist der 1-State-Prädiktor analog zum Ein-Bit-Prädiktor der Sprungvorhersage. Beispiel: Der Flur hat das Sekretariat S und das Büro vom Chef C als Nachbarräume (siehe Abbildung 5).

  Abbildung 6 zeigt den Vorhersagegraph des 1-State-Prädiktors für den Flur. Dieser ist wie folgt zu verstehen: Die Zustände geben an, welcher Raum vorhergesagt wird. Geht eine Person vom Flur in das Büro vom Chef C, wird der Prädiktor mit dem Zustand C initialisiert. Befindet sich diese Person nun wieder im Flur, wird vorhergesagt, dass sie als nächstes in das Büro vom Chef C geht. Ist dies der Fall, bleibt der Prädiktor im Zustand C. Andererseits geht die Person nun aber in das Sekretariat S, wechselt der Prädiktor in den Zustand S und sagt somit das nächste Mal das Sekretariat voraus.

- Bei *drei* Nachbarräumen muss der Ein-Bit-Prädiktor der Sprungvorhersage folgendermaßen erweitert werden. Beispiel: Der Flur hat zusätzlich das Büro vom Mitarbeiter M als Nachbarraum (siehe Abbildung 7). Abbildung 8 zeigt den Vorhersagegraph.

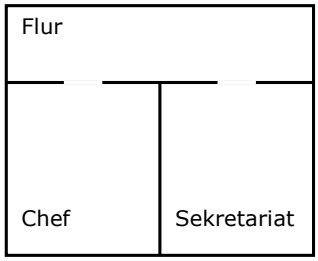


Abbildung 5: Grundriss Flur, Sekretariat und Büro Chef

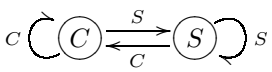


Abbildung 6: Vorhersagegraph des 1-State-Prädiktors für den Flur mit 2 Nachbarräumen

- Bei *mehr* Nachbarräumen kann das Prinzip analog fortgesetzt werden. Jeder Raum ist ein Knoten und es handelt sich immer um einen vollständigen Graphen.

Speicher- und Rechenaufwand. Speicher- und Rechenaufwand des 1-State-Prädiktor sind sehr gering. Es muss nur für jeden Raum und jede Person die ID des zu vorhersagenden Raumes gespeichert werden.

Vorteile. Ein Vorteil ist hier das schnelle Anlernen. Eine Person muss nur einmal in der Vergangenheit den aktuellen Raum besucht haben, schon kann eine Vorhersage über den nächsten Raum getroffen werden.

Nachteile. Ein möglicher Nachteil ist das zu schnelle Umlernen. Eine Person geht immer in den gleichen Raum. Wenn sie jetzt einmalig in einen anderen Raum geht, sagt der Prädiktor als nächstes den falschen Raum vorher, d.h der Prädiktor ist sehr sensibel gegenüber einmaligen Abweichungen von der Gewohnheit. Eine Abhilfe schaffen die 2- oder k-State-Kontextprädiktoren.

5.1.2 2-State-Kontextprädiktor

Es soll der Zwei-Bit-Prädiktor mit Sättigungszähler verwendet werden. Das erste Bit wird wie beim 1-State-Kontextprädiktor zum Speichern des nächsten Raumes verwendet. Das zweite Bit wird zum Wechseln zwischen den sicheren und unsicheren Zuständen eingesetzt. Unabhängig vom zweiten Bit wird also immer der im ersten Bit gespeicherte Raum vorhergesagt. Die Bezeichnung 2-State-Kontextprädiktor rührt daher, dass der Prädiktor für jeden Raum bzw. Kontext zwei Zustände vorsieht, in denen dieser Raum bzw. Kontext vorhergesagt wird.

- Ein Raum hat *zwei* Nachbarräume. Hier kann der Zwei-Bit-Prädiktor der Sprungvorhersage ohne weitere Anpassung verwendet werden. Beispiel: Der Flur hat das Büro vom Chef C und das Sekretariat S als Nachbarräume (siehe Abbildung 5).

  Abbildung 9 zeigt den Vorhersagegraph des 2-State-Prädiktors. Die Bezeichnung der Zustände besteht aus der ID eines Raumes und einem Zähler. Geht eine Person vom Flur in das Büro vom Chef C, wird der Zustand $C0$ initialisiert. Befindet sich die Person nun wieder im Flur, wird vorhergesagt, dass sie als nächstes in das Büro vom Chef C geht. Ist dies der Fall,

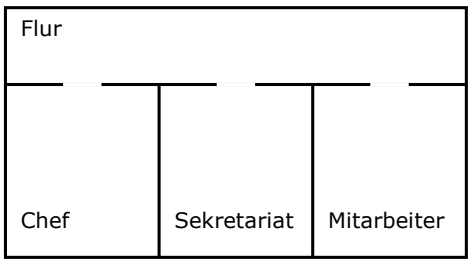


Abbildung 7: Grundriss Flur, Sekretariat, Büro Chef und Büro Mitarbeiter

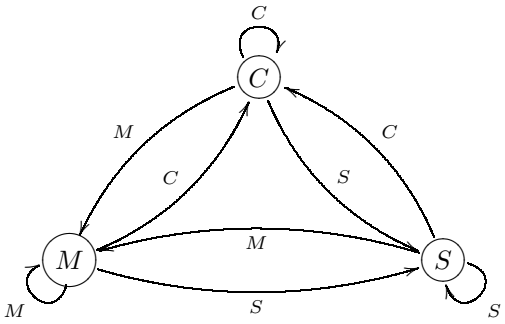


Abbildung 8: Vorhersagegraph des 1-State-Prädiktors für den Flur mit 3 Nachbarräumen

wird in den Zustand $C1$ umgeschaltet, d.h. der Zähler wird erhöht. Es wird somit das nächste Mal wieder das Büro vom Chef C vorhergesagt. Geht die Person nun irgendwann einmal vom Flur in das Sekretariat S, wird der Zustand wieder auf $C0$ gesetzt. Damit wird immer noch das Büro vom Chef vorhergesagt. Erst wenn die Person zweimal aufeinanderfolgend vom Flur in das Sekretariat S geht, wird der Zustand auf $S0$ gesetzt wird, und es wird als nächstes das Sekretariat vorhergesagt.

- Ein Raum hat *drei* Nachbarräume. Hier muss der Zwei-Bit-Prädiktor erweitert werden. Eine mögliche Erweiterung wird am folgenden Beispiel gezeigt: Der Flur hat zusätzlich das Büro vom Mitarbeiter M als Nachbarraum (siehe Abbildung 7).

  Der Vorhersagegraph in Abbildung 10 ist wie folgt zu verstehen. Die Initialisierung und der Übergang in einen sicheren Zustand ist wie oben beschrieben. Eine Person ist mehrmals vom Flur in das Büro vom Chef gegangen, der Prädiktor befindet sich also im sicheren Zustand $C1$. Wenn sie das nächste Mal vom Flur in einen anderen Raum als das Büro vom Chef geht, wechselt der Prädiktor in den Zustand $C0$, sagt also immer noch das Büro vom Chef voraus. Geht die Person nun vom Flur in das Sekretariat, schaltet der Prädiktor in den Zustand $S0$ unabhängig von dem vorher vom Flur aus betretenen Raum, und sagt somit das Sekretariat als nächstes voraus. Befindet sich der Prädiktor zum Beispiel im Zustand $C1$ und die Person betritt das nächste Mal nicht den Raum C, d.h. es wird der Raum M oder S betreten ($\neg C = M \vee S$), so geht die Information verloren, um welchen Raum es sich handelt.

- Hat ein Raum *mehr* als drei Nachbarräume kann das Prinzip analog fortgesetzt werden. Auch hier ist zu beachten, dass die Knoten mit 0 an der zweiten Stelle, d.h. die unsicheren Zustände, einen vollständigen Graphen bilden.

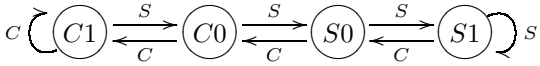


Abbildung 9: Vorhersagegraph des 2-State-Prädiktors für den Flur mit 2 Nachbarräumen

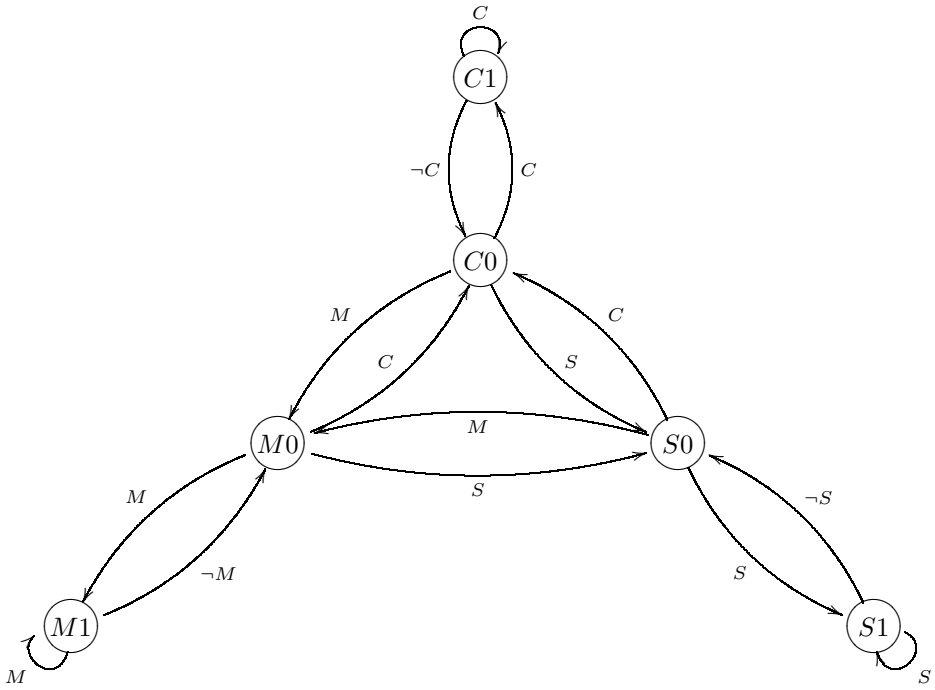


Abbildung 10: Vorhersagegraph des 2-State-Prädiktors für den Flur mit 3 Nachbarräumen

Speicher- und Rechenaufwand. Beim 2-State-Prädiktor muss zusätzlich für jeden Raum und jede Person zur ID des zu vorhersagenden Raumes der Zähler gespeichert werden, d.h. auch hier ist der Speicheraufwand sehr gering. Der Rechenaufwand ist unwesentlich größer als beim 1-State-Prädiktor.

Vorteile. Auch hier wird der Prädiktor schnell angelernt. Das Umlernen wird etwas verlangsamt, so dass einmaliges Ändern der Gewohnheit keine Auswirkung hat. Ein annähernd dauerhaftes Erlernen einer Gewohnheit mit Abweichungen, die übergangen werden sollen, kann mit einem k-State-Kontextprädiktor erreicht werden.

Nachteile. Muster im Bewegungsablauf können nicht dauerhaft erlernt werden. Bei zwei aufeinanderfolgenden Abweichungen der Gewohnheit hat das System sich die Änderung gemerkt. Dies kann in einigen Fällen zu schnell sein. Auch das Pendeln zwischen zwei Räumen kann dazu führen, dass immer der falsche Raum vorhergesagt wird.

5.1.3 k-State-Kontextprädiktor

Die ersten beiden Prädiktoren waren Spezialfälle des k-State-Kontextprädiktor. Beim k-State-Kontextprädiktor gibt es jetzt für jeden Raum k Zustände, in denen dieser vorhergesagt wird. Der k-State-Kontextprädiktor ist die Abbildung des n-Bit-Prädiktor mit Sättigung, wobei $k = 2^{n-1}$.

Das erste Bit gibt wieder den nächsten Raum an. Die weiteren $n-1$ Bits dienen zum Aufzählen. Ist der Zähler gesättigt, werden k Fehlvorhersagen benötigt, um einen anderen Raum vorherzusagen. Beispielhaft soll hier nur ein Raum mit *drei* Nachbarräumen betrachtet werden. Sei $h = k - 1$.

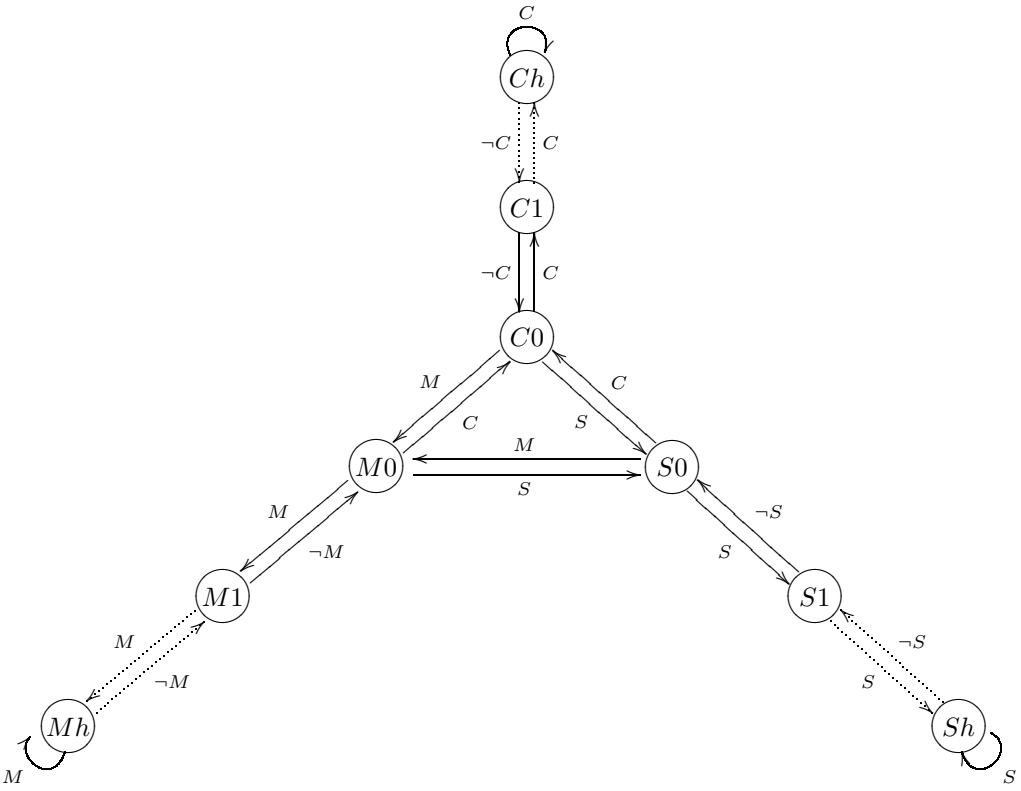


Abbildung 11: Vorhersagegraph des k-State-Prädiktors für den Flur mit 3 Nachbarräumen

Der Vorhersagegraph (Abbildung 11) ist analog zu dem des 2-State-Prädiktors zu verstehen. Der Unterschied liegt darin, dass der Prädiktor erst den sichersten Zustand erreicht hat, wenn der Zähler auf k steht. In diesem Fall werden nun k aufeinanderfolgende Übergänge in andere Nachbarräume benötigt, bis ein anderer Raum vorhergesagt wird. Der Speicheraufwand für den k-Bit-Prädiktor ist nicht oder unwesentlich größer als beim 2-Bit-Prädiktor. Auch hier muss die ID des zu vorhersagenden Raumes und der Zähler für jeden Raum und jede Person gespeichert werden.

Nach der Abbildung des 2-Bit-Prädiktor auf die Kontextvorhersage, wobei es für jeden Raum $2^{2-1} = 2$ Zustände gibt (2-State-Prädiktor), folgt der 3-Bit-Prädiktor, der für jeden Raum $2^{3-1} = 4$ Zustände liefert (4-State-Prädiktor). Da man im Bereich der Kontextvorhersagen aber der Einschränkung auf Bits nicht unterliegt, kann der Sättigungszähler nicht nur Zweierpotenzen als Wert annehmen, sondern alle natürlichen Zahlen. Somit gibt es beispielsweise auch den 3-State-Kontextprädiktor, für den es kein Urbild in der Sprungvorhersage gibt, da die Anzahl der daraus resultierenden Bits ($\log_2 3+1$) keine natürliche Zahl ist. Es gibt also nur für k-State-Kontextprädiktoren ein Urbild im Bereich der Sprungvorhersage, wenn gilt: $\log_2 k + 1 \in \mathbb{N}$.

Mit k (Sättigungszähler) wird sozusagen angegeben, wie schnell eine Gewohnheit geändert werden soll. Eine weitere Idee beim k-State-Prädiktor ist auch das Erlernen von k, d.h. es soll erlernt werden, mit welcher Geschwindigkeit eine Gewohnheit geändert werden soll.

5.2 Globale zweistufige Kontextprädiktoren

Bei den globalen zweistufigen Kontextprädiktoren wird pro Person eine Folge der zuletzt betretenen Räume betrachtet. Anhand dieser Folge wird der nächste Raum bestimmt. Die Länge der Folge wird durch die Ordnung r bestimmt, d.h. es werden die letzten r Räume zur Vorhersage herangezogen. Zur Bestimmung des nächsten Raumes werden alle möglichen Muster aufeinander folgender Räume in einer Musterverlaufstabelle abgespeichert. Zur Vorhersage des nächsten Raumes gibt es dann in der zweiten Stufe zwei Möglichkeiten, einen 2-State-Kontextprädiktor oder eine Häufigkeitsanalyse. Zur Selektierung des Musters aus der Tabelle gibt es ein Schieberegister, in dem die aktuellen abgelaufenen Räume gespeichert sind. Wird ein neuer Raum betreten, werden alle Einträge des Register nach links verschoben und der neue Raum wird rechts eingetragen.

Sei n die Anzahl der Räume und r die Ordnung, dann gibt es n^r Muster. Da aber bei den globalen zweistufigen Prädiktoren diejenigen Muster nie eintreten, bei denen zwei aufeinanderfolgende Einträge gleich sind (z.B. Flur - Sekretariat - Sekretariat), ist die Anzahl der möglichen Muster $n \cdot (n-1)^{r-1}$. Dies ist also die maximale Größe der Musterverlaufstabelle unter der Annahme, dass jeder Raum von jedem anderen direkt erreichbar ist. Da dies in der Regel nicht der Fall sein wird, reduziert sich die Komplexität in der Realität noch weiter.

5.2.1 Zweite Stufe mit 2-State-Kontextprädiktor

Hier wird in der zweiten Stufe ein 2-State-Prädiktor eingesetzt, d.h. zu jedem Muster in der Musterverlaufstabelle gibt es einen 2-State-Prädiktor der den nächsten Raum vorhersagt (siehe Abschnitt 5.1.2).

Beispiel. Es werden die drei Räume F (Flur), S (Sekretariat) und C (Büro vom Chef) betrachtet, und als Ordnung wird 3 angenommen. Dann gibt es $3 \cdot 2^2 = 12$ Muster. Es soll folgende Raumsequenz betrachtet werden:

F S C F C S F S F C S F C S

Dazu existiert dann das Schieberegister und die Musterverlaufstabelle aus Abbildung 12, anhand der vorhergesagt wird, dass als nächstes der Flur F betreten wird.

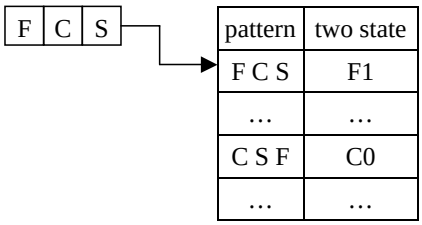


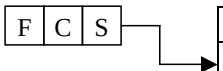
Abbildung 12: Zweistufiger Prädiktor mit 2-State-Prädiktor

Statt eines 2-State-Kontextprädiktors in der zweiten Stufe kann auch hier ein k-State-Kontextprädiktor verwendet werden, um die Sättigung des Zählers zu steuern.

5.2.2 Zweite Stufe mit Häufigkeitsanalyse

Hier wird in der zweiten Stufe die Häufigkeit des nächsten Raumes hochgezählt. Jetzt wird der Raum mit der größten Häufigkeit vorhergesagt. Dieser Prädiktor entspricht dem Markov-Prädiktor aus der Datenkomprimierung [4].

Beispiel. Es wird das analoge Beispiel wie oben betrachtet. Abbildung 13 zeigt das Schieberegister und die Musterverlaufstabelle für den zweistufigen Prädiktor mit Häufigkeiten.



pattern	next room	frequency
F C S	F	2
	C	0
…	…	…
C S F	C	1
	S	1
…	…	…

Abbildung 13: Zweistufiger Prädiktor mit Häufigkeiten

Speicher- und Rechenaufwand. Der Speicheraufwand ist abhängig von der Anzahl der Räume und von der gewählten Ordnung. Sei die Anzahl der Räume n und die Ordnung r, dann müssen für jede Person $n \cdot (n-1)^{r-1}$ Muster gespeichert werden. Werden in der zweiten Stufe 2-State-Prädiktoren eingesetzt, wird der Speicheraufwand für genau so viele 2-State-Prädiktoren wie Muster benötigt.

Wird die zweite Stufe mit Häufigkeiten umgesetzt, müssen zu jedem Muster dann die vom letzten Raum des Musters erreichbaren Räume mit den zugehörigen Häufigkeiten gespeichert werden. Sei k_i ($k_i < n$, $i = 1, \ldots, n$) die Anzahl der vom i-ten Raum aus erreichbaren Räume, dann ist die Anzahl der für jede Person zu speichernden Häufigkeiten $\sum_{i=1}^{n} (n-1)^{r-1} \cdot k_i$. Für diese Anzahl ist $n \cdot (n-1)^{r-1} \cdot (n-1) = n \cdot (n-1)^r$ eine obere Grenze.

Vorteile. Komplizierte Muster können erkannt werden. Da jedes Muster separat behandelt wird, kann es nicht zu unerwünschten **Interferenzen** [3] zwischen zwei Mustern wie in der Prozessorarchitektur kommen.

Nachteile. Wird die zweite Stufe mit Häufigkeiten umgesetzt, ist nach sehr vielen Durchläufen eines Muster ein Umlernen nur in langer Zeit wieder möglich. Wenn zum Beispiel 1000 mal nach dem Durchlaufen eines Musters der Raum R betreten wird, muss auch 1000 mal nach Durchlaufen diese Musters ein anderer Raum betreten werden, damit der andere Raum vorhersagt wird.

5.3 Lokale zweistufige Kontextprädiktoren

Bei den lokalen zweistufigen Kontextprädiktoren werden die Bewegungsabläufe nicht durch alle Räume betrachtet. Das Schieberegister enthält nicht die globale Historie, in welchen Räumen die betreffende Person zuvor war, sondern die Räume, welche die Person nach dem lokalen Raum aufgesucht hat.

Es gibt für jeden Raum und jede Person einen lokalen zweistufigen Prädiktor, d.h. es werden pro Person für jeden Raum separat die Nachfolgeräume als Muster herangezogen. Als Nachfolgeräume können nur Nachbarräume in Betracht kommen. In ihrer Funktionsweise sind die lokalen zweistufigen Prädiktoren analog zu den globalen zweistufigen Prädiktoren. Es gibt also auch wieder verschiedene Ausprägungen in der zweiten Stufe.

Beispiel. Der Flur F hat das Büro vom Chef C, das Sekretariat S und das Büro vom Mitarbeiter M als Nachbarräume. Geht nun eine Person folgendermaßen durch die Räume:

F C F S C F C M F S F

Die Person befindet sich im Flur. Das zur Vorhersage, welchen Raum die Person nach dem Flur betritt, benutzte Schieberegister, mit dem der Eintrag in der Musterverlaufstabelle selektiert wird, ist nun folgendes:

C S C S

Speicher- und Rechenaufwand. Sei r die Ordnung und sei n_s die Anzahl der Nachbarräume des Raumes s, dann muss für jede Person für jeden Raum s eine Tabelle mit $(n_s)^r$ Mustern gespeichert werden. Man beachte, dass hier im Gegensatz zu den globalen zweistufigen Prädiktoren auch der gleiche Raum in einem Muster hintereinander auftreten kann.

Vorteile. Mit einer kleineren Ordnung können hier auch längere globale Muster erkannt werden. Muster der Art „nach mehrmaligem Aufsuchen eines bekannten Raumes, wird ein anderer Raum aufgesucht“ können über einen längeren Bewegungsablauf erkannt werden.

Nachteile. Die Anlernphase ist extrem lang. Zusammenhängende Bewegungen über mehrere Räume werden nicht erkannt.

5.4 Erweiterung der zweistufigen Kontextprädiktoren

Die zweistufigen Kontextprädiktoren können analog zum *Prediction by Partial Matching* (PPM) aus der Datenkomprimierung erweitert werden. Hier wird statt der festen Ordnung in der ersten Stufe eine maximale Ordnung m gewählt. Dann wird mit dieser maximalen Ordnung m ein Muster entsprechend der letzten m Räume gesucht. Wird nun kein Muster der Länge m gefunden, wird das Muster der Länge $m-1$ gesucht, d.h. die letzten $m-1$ Räume. Wird kein Muster gefunden, kann dieser Prozess bis zur Ordnung 1 durchgeführt werden.

5.5 Überblick: Implementierte Kontextprädiktoren

Beim Abbilden der Sprungvorhersage auf die Kontextvorhersage wird ein Sprung als Übergang von einem Kontext zu einem anderen Kontext interpretiert. Dieser Kontext kann wie im Beispiel vorab der Personenkontext sein, welcher nur den Raum enthält, in dem sich die Person soeben befindet.

Tabelle 1: Implementierte Kontextprädiktoren

<table>
<tr><th>lokal</th><th>global</th></tr>
<tr><td>Für jede Person und jeden Raum werden nur die Übergänge von diesem Raum ausgehend betrachtet.</td><td>Jede Person wird durch alle Räume verfolgt.</td></tr>
<tr><td>einstufig
1-State, 2-State, k-State</td><td></td></tr>
<tr><td>zweistufig
Beispiel: Flur F mit den Nachbarräumen Büro Chef C, Sekretariat S, Büro Mitarbeiter M und Druckerraum D

Schieberegister
<table><tr><td>C</td><td>S</td><td>S</td><td>M</td></tr></table>
Musterverlaufstabelle (2-State)
<table><tr><th colspan="4">Muster</th><th>Raum</th><th>n</th></tr><tr><td colspan="4">...</td><td>...</td><td>...</td></tr><tr><td>C</td><td>S</td><td>S</td><td>M</td><td>D</td><td>0</td></tr><tr><td colspan="4">...</td><td>...</td><td>...</td></tr></table>
Musterverlaufstabelle (Häufigkeit)
<table><tr><th colspan="4">Muster</th><th>Raum</th><th>Häufigkeit</th></tr><tr><td colspan="4">...</td><td>...</td><td>...</td></tr><tr><td>C</td><td>S</td><td>S</td><td>M</td><td>D</td><td>4</td></tr><tr><td colspan="4" rowspan="2"></td><td>S</td><td>3</td></tr><tr><td>M</td><td>3</td></tr><tr><td colspan="4">...</td><td>...</td><td>...</td></tr></table>
Analog zum PAp:
P - pro Raum ein Schieberegister
p - pro Raum eine Musterverlaustabelle</td><td>zweistufig
Beispiel: Person wird durch die 5 Räume F, C, S, M, D verfolgt

Schieberegister
<table><tr><td>F</td><td>C</td><td>F</td><td>M</td></tr></table>
Musterverlaufstabelle (2-State)
<table><tr><th colspan="4">Muster</th><th>Raum</th><th>n</th></tr><tr><td colspan="4">...</td><td>...</td><td>...</td></tr><tr><td>F</td><td>C</td><td>F</td><td>M</td><td>F</td><td>1</td></tr><tr><td colspan="4">...</td><td>...</td><td>...</td></tr></table>
Musterverlaufstabelle (Häufigkeit)
<table><tr><th colspan="4">Muster</th><th>Raum</th><th>Häufigkeit</th></tr><tr><td colspan="4">...</td><td>...</td><td>...</td></tr><tr><td>F</td><td>C</td><td>F</td><td>M</td><td>F</td><td>5</td></tr><tr><td colspan="4" rowspan="2"></td><td>S</td><td>1</td></tr><tr><td>C</td><td>3</td></tr><tr><td colspan="4">...</td><td>...</td><td>...</td></tr></table>
Analog zum GAg
G - ein globales Schieberegister
g - eine globale Musterverlaufstabelle</td></tr>
</table>

6 Evaluierung

Da es für Bewegungsmuster in ubiquitären Systemen keine Benchmarks gibt, müssen synthetische Muster erstellt werden. Hier soll die Bewegung von Objekten durch ein System betrachtet werden. Folgende Anwendungen spiegeln dieses Verhalten wieder:

- Personen bewegen sich durch Räume
- Fahrt eines Aufzugs innerhalb von Stockwerken
- Bewegung durch Funkzellen im Mobilfunk

Die Evaluierung soll nun anhand von künstlichen Mustern durchgeführt werden, die die Bewegung von Personen durch Räume darstellen.

6.1 Analytischer Vergleich der einstufigen Kontextprädiktoren

Im folgendem sollen anhand von künstlichen Bewegungsabläufen der 1-State-, 2-State- und k-State-Kontextprädiktor untereinander verglichen werden. Für den k-State-Prädiktor soll beispielhaft der 4-Sate-Prädiktor herangezogen werden. Es wird wieder der Flur mit dem Büro vom Chef C, dem Sekretariat S und dem Büro vom Mitarbeiter M als Nachbarräume betrachtet. In der ersten Spalte der Tabelle 2 ist angegeben, in welcher Reihenfolge die Räume vom Flur aus betreten wurden.

Tabelle 2: Vergleich des 1-State-, 2-State- und 4-State-Prädiktors

	1-State	2-State	4-State
...C C C C C M C C C C C C C C C C C C...	2 Fehler	**1** Fehler	**1** Fehler
...C C C C C M M C C C C C C C C C C C...	**2** Fehler	3 Fehler	**2** Fehler
...C C C C C M M M C C C C C C C C C C...	**2** Fehler	4 Fehler	3 Fehler
...C C C C C M M M M M M C C C C C C...	**2** Fehler	4 Fehler	8 Fehler
...C C C C C M C M C M C M C C C C C C...	8 Fehler	**4** Fehler	**4** Fehler
...C C C C C M C C M C C M C C C C C C...	6 Fehler	**3** Fehler	**3** Fehler
...C C C C C M M C C M M C C M M C C C...	**6** Fehler	9 Fehler	**6** Fehler
...C C C C C M M C M M C M M C C C C C...	**6** Fehler	7 Fehler	7 Fehler

Aus diesen Ergebnissen ist zu erkennen, dass beim Umschalten zwischen Folgen gleicher Räume ein Prädiktor mit wenigen Zuständen besser arbeitet je länger diese Folgen gleicher Räume werden. Außerdem verringert der 4-State-Prädiktor die Fehlerrate nicht, d.h. er ist nie echt besser als der 1-State- oder 2-State-Prädiktor.

6.2 Simulation synthetischer Bewegungsmuster

Hier werden für die Evaluierung aller implementierten Kontextprädiktoren verschiedenen Bewegungsmuster simuliert. Jedes Muster besteht aus einem sich immer wiederholenden Zyklus, der so oft gelaufen wird, bis die simulierte Person 1000 Übergänge zwischen einzelnen Räumen absolviert hat. Für alle Simulationen werden die folgenden Räume verwendet: Flur F, Büro vom Chef C, Sekretariat S, Büro vom Mitarbeiter M (siehe Abbildung 7). In jeder Simulation werden alle implementierten Prädiktoren ausgewertet. Für die lokalen und globalen zweistufigen Prädiktoren wird die Ordnung 4 verwendet. In den Messungen wird für jeden Prädiktor der kumulative Fehler gemessen, d.h. die Fehlvorhersagen werden aufsummiert. Ein Prädiktor hat ein Muster erlernt, wenn die Steigung der Kurve gleich null ist. Für die Prädiktoren werden in den Diagrammen die Abkürzungen aus Tabelle 3 verwendet.

Tabelle 3: Abkürzungen der Kontextprädiktoren

L-1L-1S	lokal einstufiger 1-State-Kontextprädiktor (local 1-level 1-state)
L-1L-2S	lokal einstufiger 2-State-Kontextprädiktor (local 1-level 2-state)
L-2L-2S(4)	lokal zweistufiger Kontextprädiktor mit 2-State-Prädiktor in der zweiten Stufe und Ordnung 4 (local 2-level 2-state order 4)
L-2L-F(4)	lokal zweistufiger Kontextprädiktor mit Häufigkeitsanalyse in der zweiten Stufe und Ordnung 4 (local 2-level frequency order 4)
G-2L-2S(4)	global zweistufiger Kontextprädiktor mit 2-State-Prädiktor in der zweiten Stufe und Ordnung 4 (global 2-level 2-state order 4)
G-2L-F(4)	global zweistufiger Kontextprädiktor mit Häufigkeitsanalyse in der zweiten Stufe und Ordnung 4 (global 2-level frequency order 4)

6.2.1 Messung I

Hier soll ein Wochenablauf simuliert, d.h. Montag bis Freitag wird immer die selbe, Samstag und Sonntag eine andere Aktion durchgeführt. In dieser Simulation wird somit folgender Zyklus gelaufen:

F - S - F - S - F - S - F - S - F - S - F - C - F - C

In verkürzter Schreibweise:

$F \overset{5\times}{\rightleftharpoons} S \rightarrow F \overset{2\times}{\rightleftharpoons} C$

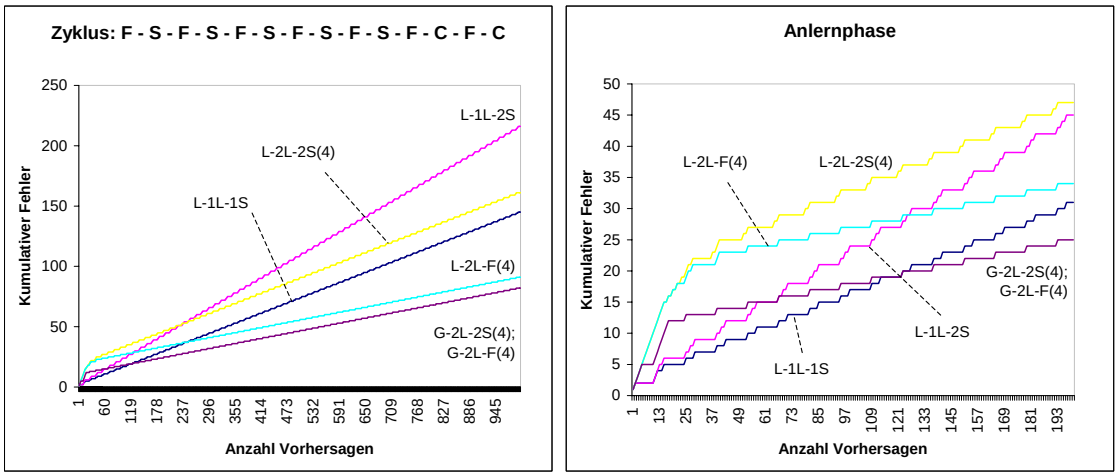


Abbildung 14: Messung I

Abbildung 14 zeigt die Ergebnisse der Simulation. Die lokalen Prädiktoren machen die Fehlvorhersagen beim Austritt aus dem Flur F. Kein Prädiktor erlernt das Muster, da für die 5 Übergänge selbst die Ordnung von 4 der lokalen zweistufigen Prädiktoren nicht ausreicht. Zum Erlernen dieses Musters wird eine Ordnung von 5 der lokalen zweistufigen Prädiktoren notwendig. Der 2-State-Prädiktor macht die meisten Fehlvorhersagen (3 pro Zyklus), und zwar jeweils 2 beim Wechsel von S auf C und 1 Fehler beim Wechsel von C auf S. Der 1-State-Prädiktor macht bei jedem Wechsel zwischen S und C 1 Fehler, also pro Zyklus 2 Fehlvorhersagen. Der lokale zweistufige Prädiktor mit 2-State-Prädiktor in der zweiten Stufe pendelt nach dem lokalen Muster S - S - S - S immer zwischen den unsicheren Zuständen von S und C hin und her, und sagt somit zweimal pro Zyklus

den falschen Raum vorher. Dieser Effekt tritt beim lokalen zweistufigen Prädiktor mit Häufigkeitsanalyse in der zweiten Stufe nicht auf. Die Häufigkeiten sind hier nach einem vollständigen Zyklus zwar immer gleich, der Prädiktor ist aber so implementiert, dass der Raum vorhergesagt wird, für den diese Häufigkeit zuerst erreicht wurde. Dieser Prädiktor macht somit nur einen Fehler pro Zyklus. Der lokale zweistufige Prädiktor mit 2-State-Prädiktor in der zweiten Stufe ist gesamt gesehen schlechter als der 1-State-Prädiktor, da das erstmalige Ablaufen der lokalen Muster in der Anlernphase mehr Fehler verursacht.

Die globalen zweistufigen Prädiktoren verhalten sich unabhängig vom 2-State-Prädiktor und der Häufigkeitsanalyse in der zweiten Stufe gleich. Sie machen beide 1 Fehler pro Zyklus, und zwar nach dem Muster S - F - S - F und vor dem Raum C. Die globalen zweistufigen Prädiktoren machen im Gesamtverlauf weniger Fehler als der lokale zweistufige Prädiktor mit Häufigkeitsanalyse, da in der Anlernphase die globalen Muster schneller gelaufen sind als die lokalen Muster.

6.2.2 Messung II

Wiederum wird ein Wochenablauf simuliert, und zwar wird Montag bis Samstag eine Aktion und Sonntag eine andere Aktion durchgeführt. Somit ergibt sich für die Simulation folgender Zyklus:

F - S - F - S - F - S - F - S - F - S - F - S - F - C

In verkürzter Schreibweise:

$F \overset{6\times}{\rightleftarrows} S \rightarrow F \overset{1\times}{\rightleftarrows} C$

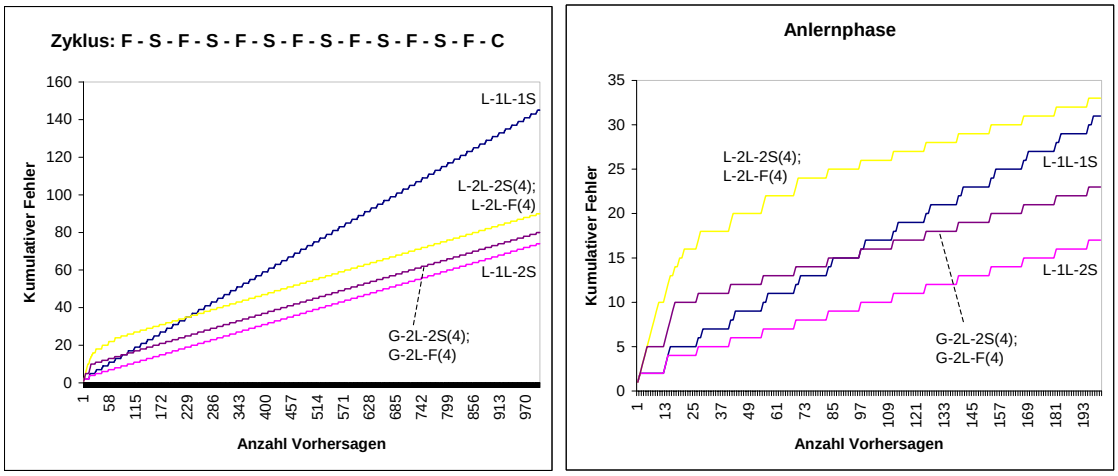


Abbildung 15: Messung II

Die Ergebnisse der Simulation zeigt Abbildung 15. Der 1-State-Prädiktor macht erwartunggemäß 2 Fehler pro Zyklus. Die anderen Prädiktoren machen alle nur eine Fehlvorhersage pro Zyklus. Der unterschiedliche Gesamtfehler tritt auf Grund der unterschiedlichen Anlerngeschwindigkeit auf. Somit hat hier aufgrund der kurzen Anlernphase der 2-State-Prädiktor den geringsten Gesamtfehler.

6.2.3 Messung III

Auch hier soll nochmals ein Wochenablauf simuliert werden, und zwar wird diesmal Montag bis Donnerstag eine Aktion und Freitag bis Sonntag eine andere Aktion durchgeführt. Somit ergibt sich für die Simulation folgender Zyklus:

F - S - F - S - F - S - F - S - F - C - F - C - F - C

In verkürzter Schreibweise:

$F \overset{4\times}{\rightleftharpoons} S \rightarrow F \overset{3\times}{\rightleftharpoons} C$

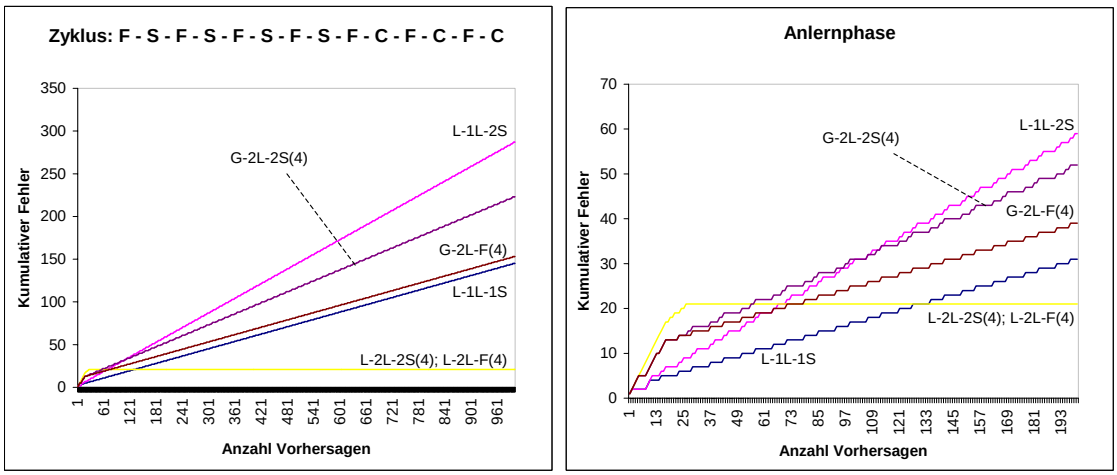


Abbildung 16: Messung III

Abbildung 16 zeigt die Simulationsergebnisse. Der 2-State-Prädiktor macht hier 4 Fehler pro Zyklus, da er sich beim Wechsel zwischen S und C immer in den sicheren Zuständen befindet. Der 1-State-Prädiktor macht wegen der zwei Wechsel pro Zyklus auch 2 Fehlvorhersagen pro Zyklus. Auch der globale zweistufige Prädiktor mit Häufigkeitsanalyse macht pro Zyklus 2 Fehler, und zwar nach den Mustern S - F - S - F und C - F - C - F. Der globale zweistufige Prädiktor mit 2-State-Prädiktor befindet nach dem Muster S - F - S - F und vor dem Raum C im sicheren Zustand für den Raum S. Er sagt somit den falschen Raum vorher und wechselt in den unsicheren Zustand von S. Beim nächsten Auftreten dieses Musters sagt der Prädiktor somit immer noch S vorher, was korrekt ist, so dass er wieder in den sicheren Zustand wechselt. Nach dem Muster C - F - C - F pendelt der 2-State-Prädiktor zwischen den zwei unsicheren Zuständen von S und C, und sagt somit zweimal den falschen Raum vorher. Er macht also insgesamt 3 Fehler pro Zyklus.

Die lokalen zweistufigen Prädiktoren lernen beide das Mustern nach einer Anlernphase, in der sie mehr Fehler machen als die anderen Prädiktoren. Aus den Messungen I-III kann die Regel abgeleitet werden, dass die Länge des Schieberegister mindestens der Anzahl der Wiederholungen im Muster entsprechen muss, damit ein solches Muster erlernt wird.

6.2.4 Messung IV

Nun sollen die Muster des Wochenablaufs verkürzt werden. Zunächst soll folgender Zyklus untersucht werden:

F - S - F - S - F - S - F - S - F - C

In verkürzter Schreibweise:

$F \overset{4\times}{\rightleftharpoons} S \rightarrow F \overset{1\times}{\rightleftharpoons} C$

Die lokalen zweistufigen Prädiktoren erlernen dieses Muster. Die globalen zweistufigen Prädiktoren und der lokale einstufige 2-State-Prädiktor machen pro Zyklus einen Fehler beim Wechsel von S auf C. Der 1-State-Prädiktor sagt pro Zyklus zweimal den falschen Raum vorher, immer beim Wechsel zwischen S und C. (Abbildung 17)

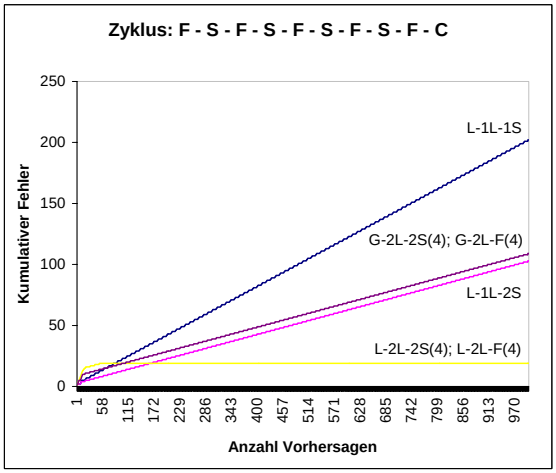


Abbildung 17: Messung IV a

Als nächstes wird folgender Zyklus simuliert:

F - S - F - S - F - C

In verkürzter Schreibweise:

$F \overset{2\times}{\rightleftarrows} S \rightarrow F \overset{1\times}{\rightleftarrows} C$

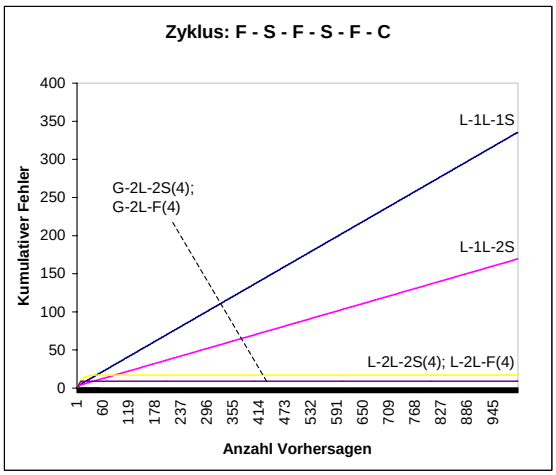


Abbildung 18: Messung IV b

Hier lernen neben den lokalen zweistufigen Prädiktoren auch die globalen zweistufigen Prädiktoren das Muster, da dieses mit Ordnung 4 auch global erfasst werden kann. Der Unterschied liegt hier wieder in der Anlernphase. Die lokalen einstufigen Prädiktoren verhalten sich wie vorab beschrieben.

6.2.5 Messung V

Hier soll untersucht werden, wie sich die Prädiktoren verhalten, wenn sich eine Gewohnheit ändert. Diese Änderung kann so aussehen, dass eine Person nach einer Folge von Aktionen immer eine

bestimmte Aktion ausführt. Nun ändert sie ihre Gewohnheit, indem sie nach dieser Folge von Aktionen einer anderen Aktion nach geht. Dabei sollen drei verschiedene Zyklen simuliert werden:

Zyklus A: Z1 - Z1 - Z1 - Z1 - Z2 - Z2 - Z2 (kurz: 4×Z1 - 3×Z2)

Zyklus B: Z1 - Z1 - Z1 - Z1 - Z1 - Z2 - Z2 - Z2 - Z2 (kurz: 5×Z1 - 4×Z2)

Zyklus C: Z1 - Z1 - Z1 - Z1 - Z1 - Z1 - Z2 - Z2 - Z2 - Z2 - Z2 (kurz: 6×Z1 - 5×Z2)

Hilfszyklus Z1: F $\overset{4\times}{\rightleftarrows}$ S $\rightarrow$ F $\overset{1\times}{\rightleftarrows}$ C

Hilfszyklus Z2: F $\overset{4\times}{\rightleftarrows}$ S $\rightarrow$ F $\overset{1\times}{\rightleftarrows}$ M

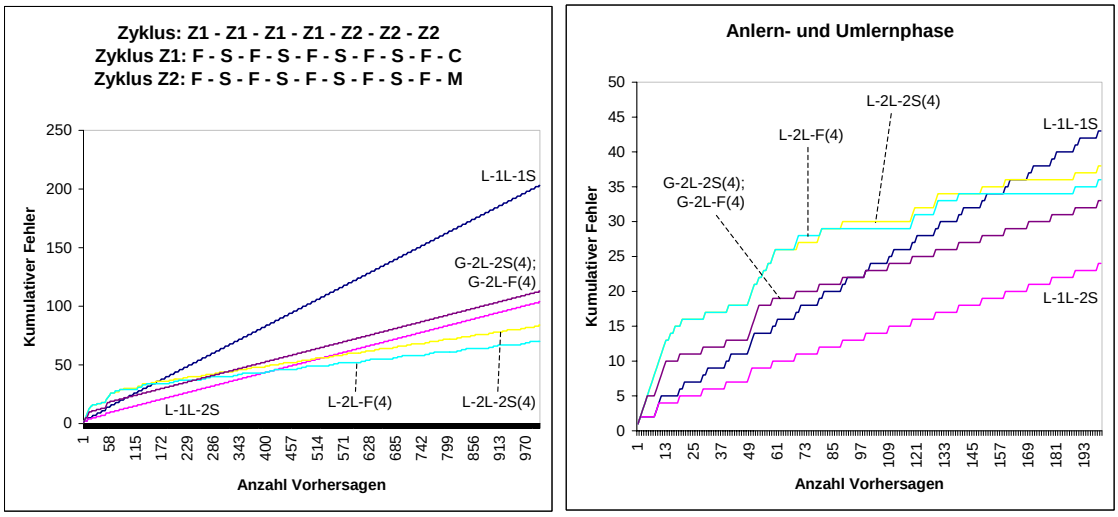


Abbildung 19: Messung V - Zyklus A

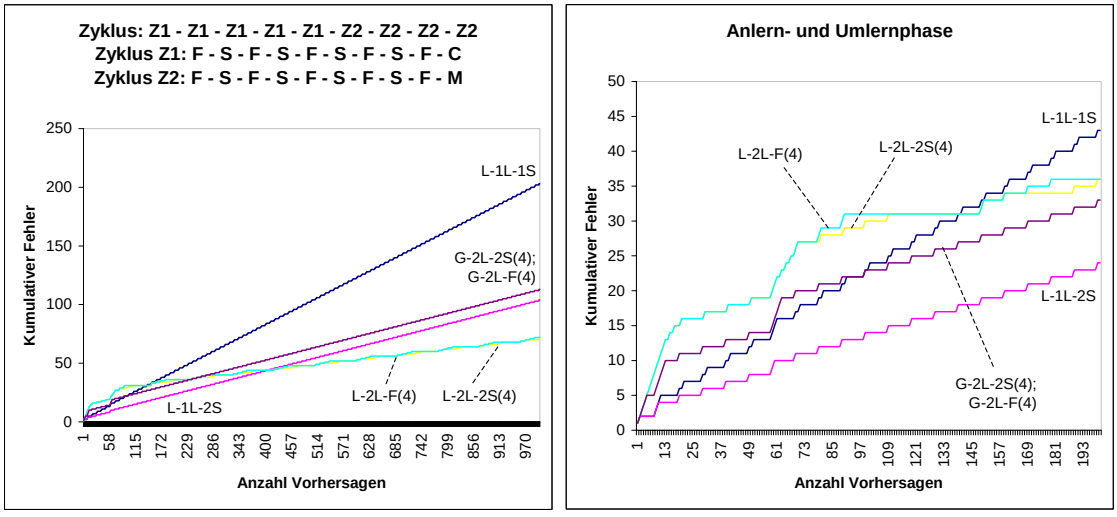


Abbildung 20: Messung V - Zyklus B

Wie die Abbildungen 19, 20 und 21 zeigen, verhalten sich die lokalen einstufigen Prädiktoren in jeder Simulation des Umlernens gleich. Sie lernen schon nach 1 bzw. 2 Änderungen eine neue Gewohnheit ohne Beachtung der Historie. Auch die globalen zweistufigen Prädiktoren zeigen keine

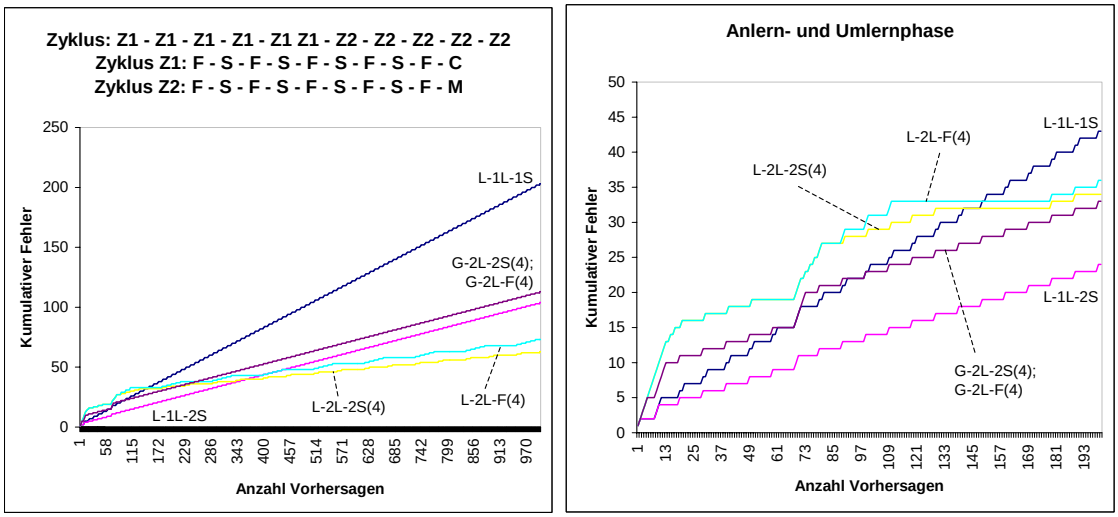


Abbildung 21: Messung V - Zyklus C

Unterschiede zwischen den Simulation, da die gelaufenen Muster global gesehen zu lang sind. Deshalb sollen hier die lokalen zweistufigen Prädiktoren näher betrachtet werden.

In der Simulation des Zykluses A macht der lokale zweistufige Prädiktor mit Häufigkeitsanalyse weniger Fehlvorhersagen. Da nach dem lokalen Muster S - S - S - S zuerst viermal nach C gegangen wird, sagt er dreimal den falschen Raum vorher, und zwar wenn nach S gegangen wird. Der lokale zweistufige Prädiktor mit 2-State-Prädiktor macht nach dem lokalen Muster S - S - S - S bei jedem Wechsel zwischen C und M 2 Fehler, d.h er sagt pro Gesamtzyklus viermal den falschen Raum vorher.

In Zyklus B verhält sich der Prädiktor mit Häufigkeitsanalyse analog wie in Zyklus A, d.h. er macht immer einen Fehler, wenn nach C gegangen wird. Dies bedeutet aber hier, er sagt viermal pro Zyklus den falschen Raum vorher. Auch der Prädiktor mit 2-State-Prädiktor macht analog wie vorab 4 Fehler pro Zyklus, d.h im Gesamtverlauf verhalten sich beide annähernd gleich. Ein kleiner Unterschied ist aber schon in der ersten Umlernphase zu erkennen. Da der Prädiktor mit 2-State-Prädiktor nach 2 Übergängen nach C diese erlernt hat, ist der kumulative Fehler zu diesem Zeitpunkt geringer als der des Prädiktors mit Häufigkeiten. Da er aber nun wieder den Übergang nach S erlernen muss, gleich sich der kumulative Fehler während der ersten Übergängen nach S in jedem Zyklus wieder an den des Prädiktor mit Häufigkeiten an, da dieser bei den Übergängen nach S nie einen Fehler macht.

In Zyklus C ist der Prädiktor mit 2-State-Prädiktor besser als der Prädiktor mit Häufigkeitsanalyse, da dieser hier 5 Fehler pro Zyklus macht, wobei der 2-State-Prädiktor wieder nur 4 Fehler macht. In der Anlernphase verhalten sie sich noch äquivalent, aber in der ersten Umlernphase zeigt sich der Unterschied schon deutlich. Diese Simulationen könnten beliebig nach diesem Schema fortgesetzt werden. Der Prädiktor mit 2-State-Prädiktor in der zweiten Stufe würde bei 4 Fehlern pro Zyklus bleiben, wogegen sich die Fehlerrate des Prädiktor mit Häufigkeitsanalyse mit der Anzahl erhöhen würde, wie oft die Hilfszyklen direkt hintereinander simuliert werden.

6.2.6 Messung VI

In dieser Simulation sollen zufällige Aktionen untersucht werden. Dazu wird folgender Zyklus herangezogen:

F - {S,C,M}

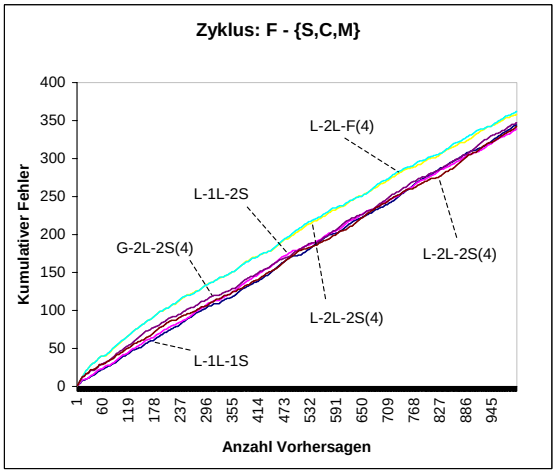


Abbildung 22: Messung VI

Werden die Übergänge zufällig gewählt (Abbildung 22), machen alle Prädiktoren gleichermaßen Fehlvorhersagen. Hierbei machen die lokalen einstufigen Prädiktoren etwas weniger Fehlvorhersagen als die globalen zweistufigen Prädiktoren, und diese machen wiederum unwesentlich weniger Fehlvorhersagen als die lokalen zweistufigen Prädiktoren. Dieses Verhalten liegt im Anlernen und Umlernen der verschiedenen Prädiktoren.

6.3 Auswertung

1-State-Kontextprädiktor. Der 1-State-Prädiktor macht bei Mustern mit langen konstanten Zyklen, d.h. es wird von einem Raum hintereinander mehrmals in den gleichen Raum gegangen, erwartungsgemäß weniger Fehlvorhersagen als bei Mustern mit Zyklen, in denen sich die nach einem Raum betretenen Räume sehr oft ändern. In den Simulationen der Wochenabläufe wird in 7 Übergängen vom Flur aus nur zweimal der Nachfolgeraum gewechselt. Wogegen in den kürzeren Zyklen in 5 bzw. 3 Übergängen vom Flur aus schon zweimal der Nachfolgeraum gewechselt wird.

In der Anlernphase zeigt der 1-State-Prädiktor die besten Ergebnisse, da er vom aktuellen Raum aus nur einen Übergang aus der Vergangenheit benötigt. Auch beim Umlernen verhält er sich analog, wird der Nachfolgeraum gewechselt, sagt er bei der nächsten Anfrage von diesem Raum aus den neuen Nachfolgeraum vorher. Dieses schnelle Umlernen ist vorteilhaft bei den schon erwähnten langen konstanten Zyklen. Hier ist der 1-State-Prädiktor besser als andere Prädiktoren (siehe Abbildung 14 und 16).

2-State-Kontextprädiktor. In den langen konstanten Zyklen (Abbildung 14 und 16) ist der 2-State-Prädiktor schlechter als der 1-State-Prädiktor, da hier pro Zyklus drei bzw. vier Fehlvorhersagen gemacht werden. In den Zyklen, in denen immer nur einmal in einen anderen Nachfolgeraum gegangen wird (Abbildung 15, 17 und 18), ist der 2-State-Prädiktor besser als der 1-State-Prädiktor. Hier macht der 2-State-Prädiktor nur einen Fehler pro Zyklus.

Das Anlernen des 2-State-Prädiktors geht auch so schnell wie beim 1-State-Prädiktor, da auch hier nach nur einem Übergang aus der Vergangenheit eine Vorhersage getroffen werden kann. Das Umlernen ist minimal langsamer als beim 1-State-Prädiktor, da zwei Fehlvorhersagen nötig sind, um den 2-State-Prädiktor aus einem sicheren Zustand zu lösen. Der 2-State-Prädiktor eignet sich als „Anlernprädiktor“ für komplexere Prädiktoren.

Lokale zweistufige Kontextprädiktoren. Da die Ordnung der lokalen zweistufigen Prädiktoren 4 ist, lernen sie die Muster aus Abbildung 14 und 15 nicht, da hier 5 bzw. 6 mal aus dem Flur in den selben Raum gegangen wird. Erst wenn nach einem lokalen Muster der Länge 4 immer ein und derselbe Nachfolgeraum auftritt, können sie das Muster erlernen. Abbildung 16 zeigt diesen Fall. In Abbildung 19, 20 und 21 erlernen sie das Muster nicht, da nach dem lokalen Muster S - S - S - S der Nachfolgeraum wechselt.

Ein Nachteil des Prädiktor mit 2-State-Prädiktor in der zweiten Stufe ist das Pendeln zwischen zwei unsicheren Zustanden (Abbildung 14). Das verhindert der Prädiktor mit Häufigkeitsanalyse in der zweiten Stufe. Hier besteht aber das Problem des langen Umlernprozesses wie Abbildungen 19, 20 und 21 zeigen. Für zwei Nachfolgeräume ist der Prädiktor mit Häufigkeitsanalyse der Grenzwert des Prädiktors mit n-State-Prädiktor für $n \to \infty$.

Beim Anlernen benötigen die lokalen zweistufigen Prädiktoren am längsten, da für eine Vorhersage jeder Raum mindesten 4 mal verlassen werden muss. Beim Umlernen gibt es zwei Fälle, zum einen wird ein neues Muster gelaufen. Hier muss analog zum Anlernen der entsprechende Raum 4 mal verlassen werden, damit eine richtige Vorhersage getroffen werden kann. Zum anderen gibt es das Umlernen nach einem bestehendem Muster. Hier hängt die Geschwindigkeit von der zweiten Stufe ab. Der 2-State-Prädiktor verhält sich hier sehr gut, wogegen der Prädiktor mit Häufigkeitsanalyse bei bestimmten Muster Probleme aufzeigt (Abbildung 21).

Globale zweistufige Kontextprädiktoren. Im Gegensatz zu den lokalen zweistufigen Prädiktoren lernen die globalen zweistufigen Prädiktoren das Muster in weniger Fällen, da hier nach einem global gelaufenen Muster sich der Nachfolgeraum zum Erlernen des Musters nicht ändern darf (Abbildung 18). Mit einer größeren Ordnung könnten auch die anderen Muster erlernt werden.

Der Unterschied zwischen dem 2-State-Prädiktor und der Häufigkeitsanalyse in der zweiten Stufe ist analog zu den lokalen zweistufigen Prädiktoren. In Abbildung 16 tritt auch das Pendeln zwischen zwei unsicheren Zuständen des 2-State-Prädiktors auf. Die Probleme des Umlernens bei der Häufigkeitsanalyse wurden nicht simuliert. Bei entsprechenden Muster ist dieses aber analog zu den lokalen zweistufigen Prädiktoren.

In der Anlernphase sind die globalen immer besser als die lokalen zweistufigen Prädiktoren, da hier die Muster der Länge 4 schneller gelaufen werden. Sie sind aber schlechter als die lokalen einstufigen Prädiktoren. Ein analoges Verhalten zeigt sich auch in der Umlernphase für den Fall, dass ein neues Muster erlernt werden muss. Andernfalls hängt das Umlernen des Raumes, der auf ein bestimmtes Muster folgt, analog zu den lokalen zweistufigen Prädiktoren wieder von der zweiten Stufe ab.

7 Zusammenfassung und Ausblick

In dieser Arbeit wurde untersucht, ob sich die Sprungvorhersagetechniken aus dem Bereich der Prozessorarchitektur zur Kontextvorhersage in ubiquitären Systemen eignen. Zwei Fragestellungen wurden dabei näher beleuchtet: Wie lassen sich diese Techniken aus der Welt der Bits auf reale Anwendungen projizieren und liefern diese Techniken in der realen Welt auch solch gute Ergebnisse.

Dazu wurden zunächst die Begriffe Kontext und Kontextvorhersage geklärt, sowie ein Anforderungskatalog an beide gestellt. Als Kern der Arbeit wurden dann die Sprungvorhersagetechniken auf die Kontextvorhersage übertragen. Dies erfolgte beispielhaft an den Bewegungsabläufen von Person in einem Gebäude. Die Evaluierung der implementierten Prädiktoren wurde mittels synthetischen Bewegungsabläufe durchgeführt, da keine realen Muster von Bewegungen durch Gebäude vorlagen. Durch die verschiedenen synthetischen Muster konnten die Prädiktoren untereinander differenziert werden. Es wurden erwartete Vor- und Nachteile aufgezeigt und belegt. Dabei wurde unter anderem festgestellt, dass die lokalen einstufigen Kontextprädiktoren sehr schnell an- bzw.

umlernen. Wogegen die die zweistufigen Kontextprädiktoren sich für komplexe Muster besser eignen. Diese beiden Vorteile könnten in einem Hybridkontextprädiktor vereint werden. Zum Beispiel kann in der Anlernphase ein 2-State-Prädiktor eingesetzt werden, der nachdem ein zweistufiger Kontextprädiktor eingelernt ist mit diesem ausgetauscht wird. Dieser Hybridprädiktor wäre in der realen Welt nützlicher, da der Mensch nicht ein halbes Jahr warten möchte bis ein System sich auf ihn eingestellt hat.

Um eine Person oder ein System nicht mit Fehlvorhersagen fehlzuleiten, sollte in Zukunft auch die Verlässlichkeit der Vorhersagen berücksichtigt werden. Das bedeutet, eine Vorhersage sollte nur dann gemacht werden, wenn einer bestimmten Wahrscheinlichkeit der korrekte Wert vorhergesagt wird. Dies kann zum Beispiel auch heißen, dass in der Anlernphase keine Vorhersage gemacht werden, da hier mit einer geringen Trefferrate zu rechnen ist.

Weitere zukünftige Aufgaben werden sein, die implementierten Prädiktoren und neue Prädiktoren an realen Bewegungsabläufen zu evaluieren. Dazu soll zum einem ein Personenverfolgungsystem verwendet werden, welches zur Zeit an der Universität Augsburg aufgebaut wird. Zum anderen soll versucht werden, die Bewegungsabläufe eines Fahrstuhls mittels der Kontextprädiktoren vorherzusehen. Ein weiterer Punkt, welcher essentiell ist für das Erlernen von menschlichen Gewohnheiten, ist das Zeitmanagement. Damit ein System sich auf den Menschen einstellen kann, muss es zum einem wissen, wann eine Aktion ausgeführt werden soll, die auf vergangenen Aktionen beruht. Zum anderen ist die Zeitspanne von großer Bedeutung, in der eine Folge von Aktionen durchgeführt wird.

A Markov-Ketten

Im folgenden soll anhand einiger Definitionen der Begriff Markov-Kette hergeleitet werden. Sei Ω der *Ereignisraum*, d.h. die Menge der *elementarer Ereignisse*.

Definition 1. Ein *stochastischer Prozess* oder *Zufallsprozess* ist eine Folge von elementaren Zufallsereignissen $X_1, X_2, ..., X_i \in \Omega, i = 1, 2, ...$

Definition 2. Die möglichen Zufallswerte in einem stochastischen Prozess heißen *Zustände* des Prozesses. Man sagt, dass sich der Prozess zum Zeitpunkt t in Zustand X_t befindet.

Definition 3. Eine *Markov-Kette r-ter Ordnung* ist ein spezieller stochastischer Prozess, bei dem zu jedem Zeitpunkt die Wahrscheinlichkeit für den nächsten Zustand nur von den vorhergehenden r Zuständen abhängt (= *Markov-Eigenschaft*), d.h. es gilt

$$\begin{aligned} &P(X_t = x_t | X_1 = x_1, X_2 = x_2, ..., X_{t-1} = x_{t-1}) \\ &\quad = P(X_t = x_t | X_{t-r} = x_{t-r}, ..., X_{t-1} = x_{t-1}) \end{aligned}$$

Oder anders ausgedrückt: der Prozess vergisst frühere Ereignisse.

Bemerkung. Oft wird mit dem Begriff Markov-Kette nur die Markov-Kette erster Ordnung gemeint, hierbei hängen dann zu jedem Zeitpunkt die Wahrscheinlichkeiten für zukünftigen Zustände nur vom momentanen Zustand abhängen, hier gilt somit

$$\begin{aligned} &P(X_t = x_t | X_1 = x_1, X_2 = x_2, ..., X_{t-1} = x_{t-1}) \\ &\quad = P(X_t = x_t | X_{t-1} = x_{t-1}) \end{aligned}$$

Eine Markov-Kette r-ter Ordnung über einer Menge S von Zuständen ist äquivalent zu einer Markov-Kette erster Ordnung über einer Menge S^r von r-Tupeln aus S. Dies folgt direkt aus

$$\begin{aligned} &P(X_{t-r+1}, ..., X_{t-1}, X_t | X_{t-r}, ..., X_1) \\ &\quad = P(X_t | X_{t-r}, ..., X_{t-1}) \end{aligned}$$

Beispiel. Sei $S = \{0, 1\}$ die Menge von Zuständen. Betrachtet man auf diesen Zuständen eine Markov-Kette zweiter Ordnung, dann ist eine Sequenz dieser Kette äquivalent zu einer Sequenz von Paaren aus einer Markov-Kette erster Ordnung (Fig.23). Die Sequenz 01101 beispielsweise entspricht $01 - 11 - 10 - 01$.

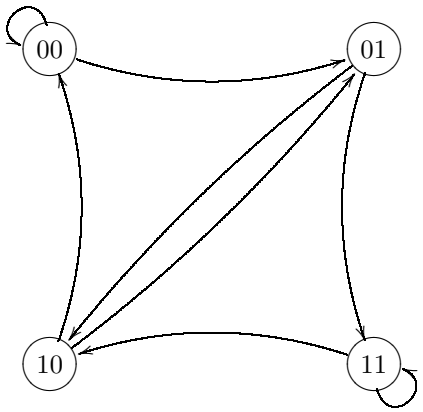


Abbildung 23: Markov-Kette erster Ordnung für 2-Tupel mit 4 Zuständen

In dieser Kette sind nicht alle Übergänge erlaubt, da auf ein Symbol jeweils nur zwei andere folgen dürfen. Auf den Zustand 01 können beispielsweise nur die Zustände 11 und 10 folgen, sonst wäre keine Äquivalenz zu einer Kette zweiter Ordnung möglich.

Obwohl sich alle Markov-Ketten r-ter Ordnung auf Markov-Ketten erster Ordnung zurückführen lassen, ist es in der Anwendung manchmal praktischer, das Modell der höheren Ordnung zu benutzen. Theoretisch können sie aber immer gleich behandelt werden.

Definition 4. Die Übergangswahrscheinlichkeiten einer Markov-Kette können in einer Übergangsmatrix dargestellt werden. Betrachte eine Markov-Kette mit n Zuständen $z_1, ..., z_n$. Sei $p_{ij}(t)$ die Übergangswahrscheinlichkeit von Zustand z_i nach Zustand z_j zum Zeitpunkt t, dann sieht die Übergangsmatrix wie folgt aus:

$$P(t) = \begin{pmatrix} p_{11}(t) & ... & p_{1n}(t) \\ ... & ... & ... \\ p_{n1}(t) & ... & p_{nn}(t) \end{pmatrix}, \quad p_{ij}(t) \geq 0, \quad \sum_{j=1}^{1} p_{ij}(t) = 1, i = 1, ..., n$$

Eine Matrix mit diesen Eigenschaften wird auch *stochastische Matrix* genannt.

Definition 5. Sind die Übergangswahrscheinlichkeiten zu jedem Zeitpunkt gleich, d.h. unabhängig von t, wird von einer *(zeitlich) homogenen Markov-Kette* gesprochen.

Eine Markov-Kette bezeichnet man als *inhomogen,* wenn sich die Übergangswahrscheinlichkeiten periodisch ändern, d.h. die Übergangswahrscheinlichkeiten hängen von der Position i in der Sequenz und einer Periodenlänge T ab.

Inhomogene Markovketten könnte man so interpretieren, dass man eine Reihe von T verschiedenen Markovketten mit gleichen Zuständen, aber unterschiedlichen Übergangswahrscheinlichkeiten betrachtet. Jedesmal, wenn ein Zustand gewechselt wird, wechselt man auch zur Markov-Kette in der Reihe. Nach dem T-ten Übergang wechselt man wieder zur ersten Kette.

Nun ergibt sich die Fragestellungen, ob man die zweistufigen adaptiven Branch Predictoren durch nicht homogene Markov-Ketten mathematisch formulieren kann?

Literatur

[1] Bagci, F., J. Petzold, W. Trumler und T. Ungerer: *Einsatz von XML zur Kontextspeicherung in einem agentenbasierten ubiquitären System*. In: *XMIDX-Workshop*, Berlin, 17.-18. Februar 2003.

[2] Beigl, M.: *Context Aware Computing*. In: *Tutorial at ARCS 2002*, Karlsruhe, April 2002.

[3] Brinkschulte, U. und T. Ungerer: *Mikrocontroller und Mikroprozessoren*. Springer-Verlag Berlin Heidelberg, 2002.

[4] Chen, I.-C. K., J. T. Coffey und T. N. Mudge: *Analysis of Branch Prediction via Data Compression*. In: *ASPLOS VII*, S. 128–137, Cambridge, Massachusetts, USA, Oktober 1996.

[5] Dix, A., T. Rodden, N. Davies, J. Trevor, A. Friday und K. Palfreyman: *Exploiting space and location as a design framework for interactive mobile systems*. In: *ACM Transaction on Computer-Human Interaction (TOCHI)*, S. 285–321, 1999.

[6] Gellersen, H. W., A. Schmidt und M. Beigl: *Multi-Sensor Context-Awareness in Mobile Devices and Smart Artifacts*. In: *Mobile Networks and Applications 7*, S. 341–351, 2002.

[7] Kaowthumrong, K., J. Lebsack und R. Han: *Automated Selection of the Active Device in Interactive Multi-Device Smart Spaces*. In: *Workshop at UbiComp'02: Supporting Spontaneous Interaction in Ubiquitous Computing Settings*, Göteborg, Schweden, 2002.

[8] Katsiri, E.: *Principles of Context Inference*. In: *Adjunct Proccedings UbiComp'02*, Göteborg, Schweden, 2002.

[9] Mozer, M. C.: *The Neural Network House: An Environment that Adapts to its Inhabitants*. In: *AAAI Spring Symposium on Intelligent Environments*, S. 110–114, Menlo Park, CA, USA, 1998.

[10] Pan, S.-T., K. So und J. T. Rahmeh: *Improving the Accuracy of Dynamic Branch Prediction Using Branch Correlation*. In: *ASPLOS V*, S. 76–84, Boston, USA, April 1992.

[11] Salber, D., A. K. Dey und G. D. Abowd: *The Context Toolkit: Aiding the Development of Context-Enabled Applications*. In: *CHI'99*, S. 434–441, Pittsburgh, PA, USA, 15.-20. Mai 1999.

[12] Schmidt, A., M. Beigl und H.-W. Gellersen: *There is more to Context than Location*. In: *Computer & Graphics Journal*, Bd. 23, S. 893–902, Dezember 1999.

[13] Yeh, T.-Y. und Y. N. Patt: *Alternative Implementation of Two-Level Adaptive Branch Prediction*. In: *ISCA-19*, S. 124–134, Gold Coast, Australia, Mai 1992.